FFDC API Style Guide and Standards Manual

IMPORTANT! Technical writers should consult their product's process guide prior to updating API reference content in order to ensure development check-in/check-out protocols are clear.

This API Style Guide and Standards Manual provides the information needed to write, edit, and maintain documentation for the FFDC API platform. Using this guide ensures API documentation is easy to understand and support.

- Reference Content Standards
- Style Guide
- Accepted Markdown Schema

Reference Content Standards

This section explains the components that need to be documented for each endpoint, and the writing and formatting standards expected for each API component.

IMPORTANT! This document is a work in progress. Please email Sarah Kittrell with suggestions and updates.

Resource Collection Description

Each group of endpoints is organized by a developer-determined group that limits the number of endpoints displayed in the API user interface. The description for this group should provide a high-level overview of the endpoints a developer can expect to find in a Resource Collection.

- Always start the description with, "Endpoints for…".
- Use complete sentences.
- Do not limit your description in a way that would require you to make revisions.

Note: While some APIs may choose to start with a simple present tense verb, this creates an issue when new endpoints are added that use different methods. For example, "Sends and receives payment information," must be revised if a DELETE endpoint is added. Once all Methods are used in a Resource Collection, it would necessitate updating each description with, "Creates, receives, updates, and deletes payment information," which is both wordy and creates unnecessary detail.

Examples

Accounts

Endpoints for account-level information available for all accounts across the financial institution and individually for selected accounts.

External Accounts

Endpoints for accounts managed by the financial institution but controlled by an external service, such as a credit card processor.

Utilities

Endpoints for access token management and other calls that assist with technical requirements used to manage the API service.

Endpoint Title

Each endpoint title must be unique to the Resource Collection and clearly describe the business function performed by the call.

- Titles should be capitalized according to current heading conventions.
- Follow the CRUD method for title creation, in which the endpoint's Method determines the verb used at the beginning of an endpoint title.

Method	Corresponding Verb	Example
GET	Retrieve or Search for	Retrieve Deposit Accounts by Account Type Search for a Deposit Account
DELETE	Delete	Delete Bill Memo for a Loan Account
POST	Create New	Create New Combined Statement
PATCH	Update	Update Phone Number for a Customer/Member
PUT	Create New or Update	Create New Loan Account Update Rate Index for a Financial Institution

Note: Search for should be used only when parameters are included with a GET that intend for the call to return only one record. GET endpoints that are designed to return a list of records, such as a list of branches for an institution, should always use Retrieve.

Exceptions

In some cases, the business function does not explicitly match the method used to perform the function. In those instances, choose the corresponding verb that best represents the business function, NOT the method.

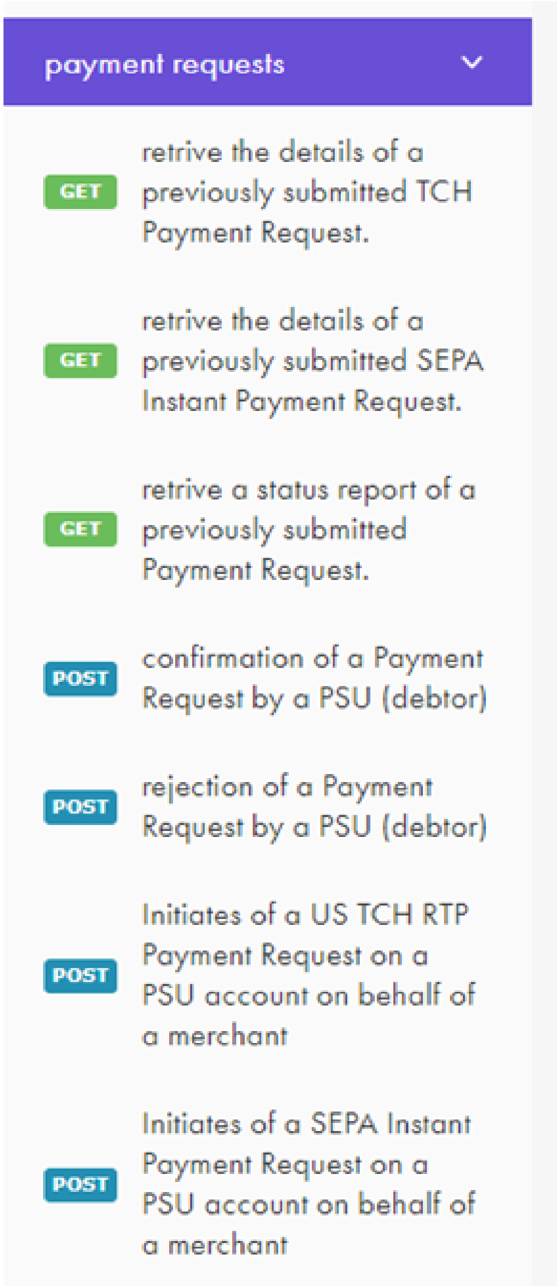
1. Creation of a new account is ***always*** referred to as origination for your product. In this case, the endpoint title should forgo the CRUD method and follow the product standard:
    - Deposit Account Origination
    - Loan Account Origination
    - Safe Deposit Account Origination
2. In order to add a custom field to the financial statement setup windows in your product, the field must first be created in a master custom field table. Then, the field is copied over to the custom field table for financial statements. In order to create the endpoint, the developer uses POST (Create New) since the record is being created in the sub-table.

However, from a user perspective, the field must already exist, and they are merely assigning the custom field to display for financial statements.

- Assign a Custom Field to Financial Statements
- Assign Custom Field Values to a Custom Field

Note: Be sure to call out the dependencies that require the naming exceptions in the endpoint description. For the second example, the endpoint description would include, "The custom field must already be created in the master custom field table in order to assign the field using this call."

Examples



payment requests

GET retrive the details of a previously submitted TCH Payment Request.

GET retrive the details of a previously submitted SEPA Instant Payment Request.

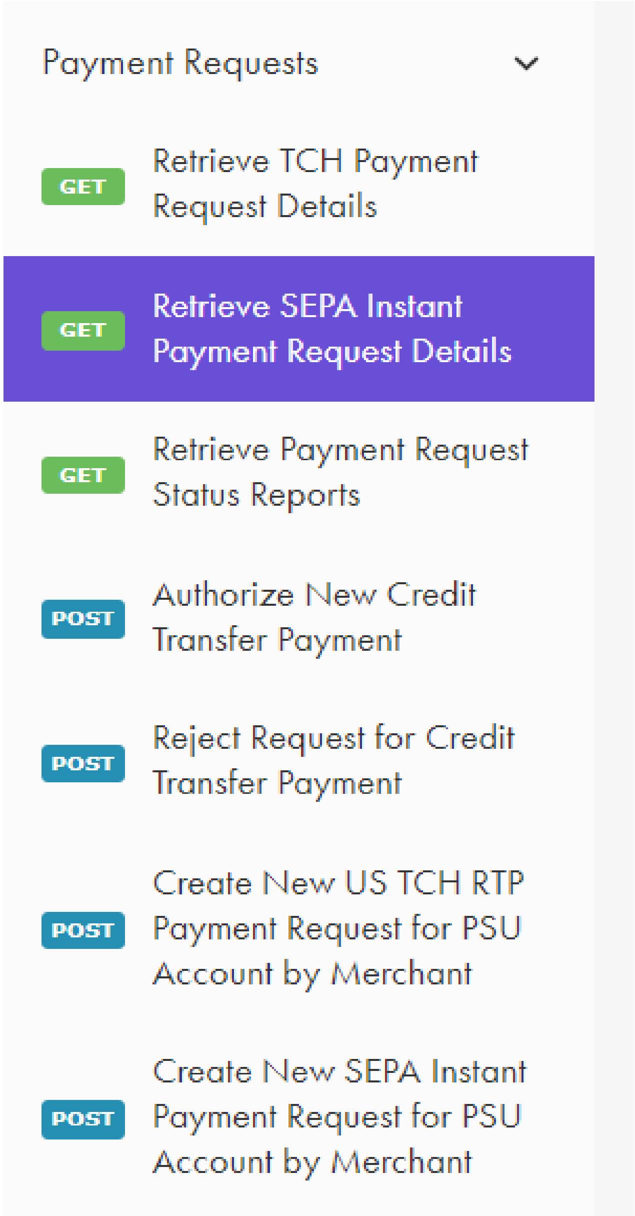
GET retrive a status report of a previously submitted Payment Request.

POST confirmation of a Payment Request by a PSU (debtor)

POST rejection of a Payment Request by a PSU (debtor)

POST Initiates of a US TCH RTP Payment Request on a PSU account on behalf of a merchant

POST Initiates of a SEPA Instant Payment Request on a PSU account on behalf of a merchant



Payment Requests

GET Retrieve TCH Payment Request Details

GET Retrieve SEPA Instant Payment Request Details

GET Retrieve Payment Request Status Reports

POST Authorize New Credit Transfer Payment

POST Reject Request for Credit Transfer Payment

POST Create New US TCH RTP Payment Request for PSU Account by Merchant

POST Create New SEPA Instant Payment Request for PSU Account by Merchant

Endpoint Description

Each endpoint description provides a few sentences that explains the purpose and functionality of an endpoint. Focus on required details or special considerations used to execute the call correctly.

- Always start the description with, "This call…".
- Use complete sentences.
- Again, the method determines the verbs used in the description.

Method	Example
GET	This call **returns** deposit accounts by account type. This call **searches for** an individual deposit account by the account number.
DELETE	This call **deletes** an existing bill memo for a loan account.
POST	This call **creates** a new combined statement for a selected customer/member.
PATCH	This call **updates** the primary phone number that is currently assigned to a customer/member.`
PUT	This call **creates** a new loan account. *or* This call **updates** the rate index applied across the financial institution that is then applied to loan accounts originated after the effective date of the rate index.

Content Recommendations

Review the following guidelines when creating endpoint descriptions to ensure adequate detail is provided:

- Describe how the system handles data included in the call:

  Backdating of new loan accounts is supported by the effectiveDt tag.
- Conditional responses:

  If checkNo is supplied, the response includes whether or not a stop/hold has been placed on the check number/amount.

  The system calculates the term and period of the account if these values are not provided in the request.
- Special notes for when another call should be used:

  Only the subject text is returned with this transaction. To retrieve the full message details for a specific message, use Correspondence Details.

Call Outs

Enter a break tag <br/>, then use the following bolded terms to call out properties that may apply to an endpoint. You ***do not*** need to add the call out if defined as None or Null.

Call Out Term	Definition
Group	Provide a group restriction when one is defined for an endpoint: Trusted Services only Employees only
Availability	Defines when the endpoint can be run. For some products, endpoints may be limited to run only when the server is in a specific mode. Your product's specifications and standards determine when the call out should be used. Use clear language that can be understood across product offerings.
Legacy Equivalent	Defines the name of a similar API call or other middleware process that was previously offered by your product.

Examples

Retrieve Closing Balance Totals for Deposit Account

This call returns the closing balance and penalty amount for a deposit or time account.

Availability: Only available during Day Mode. This transaction is processed real-time.

Legacy Equivalent: 13110 - Deposit Account Closing Balance Inquiry

Retrieve Loan History Transactions

This call returns transaction history for all loan accounts at a financial institution, including transaction amount, dates, description, and other related details.

One or more records may be returned based on the search filters added to the call. When no filter is supplied, all transactions are returned.

> ***Note:*** In this example, the ability to return all transactions is called out emphatically as it may place a strain on the institution's server ***and*** the amount of data returned to the developer may be unmanageable.

Safe Deposit Box Account Origination

This call creates a new safe deposit box (SDB) account for the selected customer/member. The account number is returned in the response in both nonencrypted form (acctNo, acctType) and encrypted form (encId).

Availability: Available only during Day Mode. This transaction is processed real-time.

Legacy Equivalent: DB_ACTION - Create New Safe Deposit Account

Update Service Description for a Loan Account

This call updates the account description for a selected loan account that is defined by the external service through which the loan account was configured.

Group: Trusted Services only

Legacy Equivalent: Update 10020 - OutputType 1

Model Description (Objects and Arrays)

Each group of request or response parameters that is nested inside a Model requires an overarching description that provides insight on what tags are contained inside. This allows the developer to decide whether they need to expand the group and review the tags in the API user interface.

- Use complete sentences.

Note: When the request body is unique to an endpoint, the model description for the request body should reflect its unique purpose.

Examples

request >	Passes the debit transfer details for the call.
messages >	Data that the server produced while processing the call.
pagingInfo >	Page data with context about the amount of tags returned from the database.
responseInfo >	Technical details returned from the sender, including the status of the server, a unique id for the call (logPtid), and confirmation of how the call was posted (memo, real-time, or future dated).
result >	Unique values returned by a call.

Parameter Description

Reminder! Parameters can be included in multiple places in an endpoint definition:

Parameter Type	Location	Purpose	Example
Path Parameter	In the URL (usually between curly braces or after a colon)	Identify a unique resource, such as and individual customer or account.	http://apiservice.com/loanpymt-send/**{custnbr}** Passes the encrypted customer/member number defined by the financial institution.
Header Parameter	In the HTTP header between the <HEAD> </HEAD> tags.	Can identify a unique resource, sort a response, or allow for	**Content-Length:** 2143

Parameter Type	Location	Purpose	Example
		filtering what response data is returned.	Indicates the maximum number of bytes that can be provided in the request.
Query Parameter	In the URL after a question mark (or combined with ampersands)	Sort or filter data returned in the response.	http://apiservice.com/loanpymt-send/?**orderBy=acctNo:asc** Sorts results by a selected tag in either ascending (asc) or descending (desc) order. **Syntax:** fieldName:order
Request Parameter	In the Request Schema	Send data to the API service.	**createDt** Defines the date on which the new index rate takes effect, and updates the Status to Active.
Response Parameter	In the Response Body	Return the data sent from the API service.	**hotCard** Indicates whether the ATM card has been marked as stolen or otherwise fraudulent.

Descriptions of individual parameters are more varied in both content and structure than other style conventions, as these descriptions rely heavily on how they can be interpreted by external developers.

Note: Use **tag** to refer to a parameter in client-facing content. Developers will probably continue to use parameter, but it is unwieldy and does not create readable help information.

- Always start the description with a 3rd person, present tense verb that describes a concrete action performed by the parameter or the content contained in the tag:
  - **Passes** the access token required to authorize the call.
  - **Defines** the employee responsible for servicing an account by their unique ID number.
  - **Filters** results by the account status.
  - **Filters** results to include or exclude account balances in the call response.
  - **Sorts** results by the selected field name.
  - **Limits** the number of records returned.
- Avoid complete sentences as they create repeated content.
- Capitalize the first word in the description. Many times, this is overlooked when the description is only a few words.
- Use relevant punctuation in the description.

IMPORTANT! Do not modify the parameter name. Update descriptions only.

Content Recommendations

Review the following guidelines when creating parameter descriptions to ensure adequate detail is provided:

- You may need to consult existing documentation to expand parameter details. Keep in mind the external developer may not have much knowledge of the application behind the API service.
- By default, a parameter description from the developer may only include:

  > Filters results by origin.

  The best practice would be to review existing client-facing documentation to see if you can add clarity to the description. For this example, you might update the description to display:

  > Filters results by the numeric code that identifies the origin of an external file type, such as 4 (ACH) or 5 (ATM).

  With this change, the external developer can contact Finastra with some basic knowledge of what they need, in this case, the origin ID that is assigned to specific transactions related to their service.

- Enter accepted values. If you need to define the values, create a bulleted list using Markdown.

  Note that external developers may not be well versed on specific tag options, so you may need to add these definitions even if they were not previously included in your legacy documentation.

  For example, a client using the product application can simply click a drop-down field to view available options. This is not available in your API unless a specific **Enumerations** endpoint (or parameter option) has been created for the field.

- Typically, you can skip defining financial terms, but vague values, such as P must be defined.

  Undefined (use the Swagger enum option)

  – enum: NSF, UCF, Deposit Rejected, Recon Rejected

  – enum: Any valid branch number

  Defined

  Values:

  - P: Process the transaction immediately.
  - M: Mark the transaction to be processed later.

- When no default is defined, no additional content is needed unless there is a request by stakeholders to add additional information about what happens when the tag is left blank.

  For example, Defines the account type for a new deposit account. When this tag is left blank, the financial institution determines the account type by the external service that submitted the call, and returns the type in the response.

Specific to Parameter Types

- For Path parameters: External developers must be able to understand what information needs to be passed in the URL to let the API service know which customer, account, or other type of record is being identified in the path parameter. Typically, developers will include a sample of what is passed or a list of values that can be passed in the parameter description.

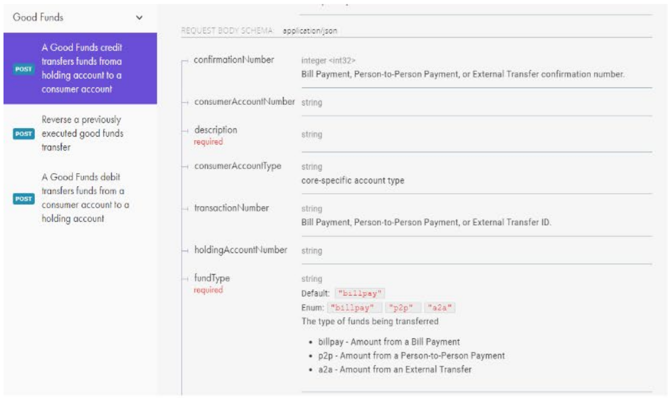
IMPORTANT! Path parameters are always required, so you must pay close attention to these descriptions.

- For Header, Query, and Request parameters: A default value may be defined by the API service that needs to be documented in the parameter description.
  - The simplest way to mark a default option is to add **(default)** after the value.
  - Alternately, you may choose to make another line break and add **Default:** with the value that is automatically passed when a tag is left blank.

Examples

Path Parameters	Description
resourceId	Passes an encrypted unique ID that identifies the report to return based on the date generated and the unique customer ID. {"rptNo":"string","createDt":"string",”custNbr:”string”}
formatId	Passes a value that determines the ordering and data format of the information returned to ensure data is configured correctly for a third-party printer. Values: • 1: Name, Mailing Address, Phone, Statement • 2: Mailing Address (name included), Statement • 3: Name, Phone, Statement (electronic delivery) • 4: Statement only
Sort/Filter Parameters	Description
descending	Sorts results in descending order by date.
offset	Filters the number of records to return starting after the record number provided.
count	Limits the maximum number of records that can be returned.
page	Filters results to return only results on the selected page index. The number of records per page is determined by the value of the count tag.
includeClosed	Indicates whether results include accounts where the Status is Closed. Values: Y, N (default)
Request/ Response Parameters	Description
atmSwitchId	Defines the ID of the ATM switch or proprietary ATM unit. Values: Other (Non Proprietary ATM), null (Proprietary ATM)
hitOption	Defines how the system validates and processes a stop payment. Values:

	• A: Same amount only • B: Both the check number and the amount • C: Same check number only • E: Either the check number or the amount of the item • O: Company ID only
limitFromCheckNo	Defines the lowest check number for an account that determines if a stop payment should be placed on an incoming transaction. If the account includes a range of checks, the value entered must be the beginning (or lowest) check number to ensure the stop payment is recorded accurately.
limitToCheckNo	Defines the highest check number for an account that determines if a stop payment should be placed on an incoming transaction. If the account includes a range of checks, the value entered must be the ending (or highest) check number to ensure the stop payment is recorded accurately.
effectiveDt	Defines the date that the Status of the additional title was updated to Active.
createDt	Defines the date the additional title was originally added to the system.
noDaysPurge	Defines the number of days after which Closed accounts tied to this account type is eligible to be purged from the database.
passbookUpdated	Indicates whether the passbook was updated for this transaction as part of the request. Values: • Y: Teller or external system has updated the passbook. • N: Teller or external system has not updated the passbook, but a passbook is involved in the transaction. • NULL: Not a passbook transaction or the account isn't associated with a passbook.


Good Funds
POST A Good Funds credit transfers funds froma holding account to a consumer account
POST Reverse a previously executed good funds transfer
POST A Good Funds debit transfers funds from a consumer account to a holding account

REQUEST BODY SCHEMA: application/json

confirmationNumber integer <int32>
Bill Payment, Person-to-Person Payment, or External Transfer confirmation number.

consumerAccountNumber string

description required string

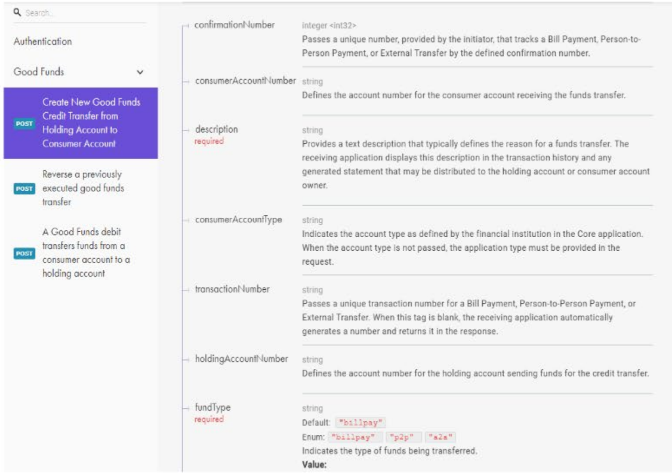
consumerAccountType string
core-specific account type

transactionNumber string
Bill Payment, Person-to-Person Payment, or External Transfer ID.

holdingAccountNumber string

fundType required string
Default: "billpay"
Enum: "billpay" "p2p" "a2a"
The type of funds being transferred

- billpay - Amount from a Bill Payment
- p2p - Amount from a Person-to-Person Payment
- a2a - Amount from an External Transfer


Search...
Authentication
Good Funds
POST Create New Good Funds Credit Transfer from Holding Account to Consumer Account
POST Reverse a previously executed good funds transfer
POST A Good Funds debit transfers funds from a consumer account to a holding account

confirmationNumber integer <int32>
Passes a unique number, provided by the initiator, that tracks a Bill Payment, Person-to-Person Payment, or External Transfer by the defined confirmation number.

consumerAccountNumber string
Defines the account number for the consumer account receiving the funds transfer.

description required string
Provides a text description that typically defines the reason for a funds transfer. The receiving application displays this description in the transaction history and any generated statement that may be distributed to the holding account or consumer account owner.

consumerAccountType string
Indicates the account type as defined by the financial institution in the Core application. When the account type is not passed, the application type must be provided in the request.

transactionNumber string
Passes a unique transaction number for a Bill Payment, Person-to-Person Payment, or External Transfer. When this tag is blank, the receiving application automatically generates a number and returns it in the response.

holdingAccountNumber string
Defines the account number for the holding account sending funds for the credit transfer.

fundType required string
Default: "billpay"
Enum: "billpay" "p2p" "a2a"
Indicates the type of funds being transferred.
Value:

Style Guide

Balance

When you end your sentence with a verb clause, it causes the reader to think the sentence is unbalanced. Try to avoid creating sentences where the verb clause is far away from the main subject.

- Correct: Transaction data is returned for all accounts.
- Incorrect: Transaction data for all accounts is returned.

Verbs

Follow current standards that require the use of present tense verbs.

- Correct: This call **replies** to a message based on the message ID.
- Incorrect: This call will reply to a message based on the message ID.

Accepted Markdown Schema

There are two accepted forms of formatting marks for REST API:

- **Markdown** is the preferred way to format in the code line.
- **Inline HTML** should be reserved for when formatting is needed that is not supported in markdown schema.

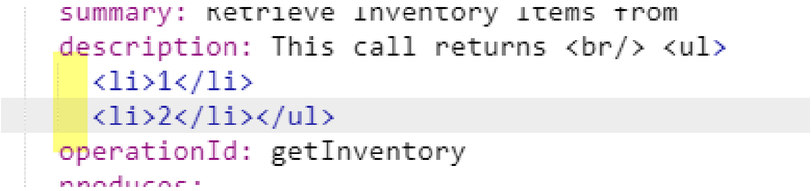
IMPORTANT! Markdown can't be used inside HTML tags, such as bolding a word inside an HTML-formatted table. Use HTML tags to bold the word inside the table instead.

This guide includes direct examples to apply uniform formatting marks throughout API documentation.

Note: The full markdown schema is not supported at this time. Refer to the table below for accepted formatting.

Name	Markdown	Description	Output
Strong emphasis (bold)	**Availability:**	Two asterisks before and after the area to be bolded.	**Availability:**
Line break	Test <br/> Test Use the HTML convention of a break tag.	Start a new line below the current line. ***Note:*** Markdown performs this action when you add an extra line space between lines, but Swagger does not support full markdown.	Test Test

Bulleted List	Values:
	<ul>
	  <li>P: First priority</li>
	  <li>M: Middle priority</li>
	  <li>L: Lowest priority</li>
	</ul>
		Markdown schema not supported for bulleted lists.	• P: First priority
			• M: Middle priority
			• L: Lowest priority

Note: Keep in mind that when you press Enter to start your list, you must also press Tab to indent the following line. Do this once after the first line, and then press Enter for all subsequent lines.		 summary: Retrieve inventory items from description: This call returns <br/> <ul> <li>1</li> <li>2</li></ul> operationId: getInventory produces:	