Katana Developer Guide

Document version 237.

Notice

THE INFORMATION CONTAINED HEREIN IS PROVIDED TO RECIPIENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTY OF TITLE OR NON- INFRINGEMENT. ALL SUCH WARRANTIES ARE EXPRESSLY DISCLAIMED.

HEARTLAND PAYMENT SYSTEMS SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, WHETHER RESULTING FROM BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, OR OTHERWISE, EVEN IF HEARTLAND PAYMENT SYSTEMS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. HEARTLAND PAYMENT SYSTEMS RESERVES THE RIGHT TO MAKE CHANGES TO THE INFORMATION CONTAINED HEREIN AT ANY TIME WITHOUT NOTICE.

THIS DOCUMENT AND ALL INFORMATION CONTAINED HEREIN IS PROPRIETARY HEARTLAND PAYMENT SYSTEMS INFORMATION. UNDER ANY CIRCUMSTANCES, RECIPIENT SHALL NOT DISCLOSE THIS DOCUMENT OR THE SYSTEM DESCRIBED HEREIN TO ANY THIRD PARTY WITHOUT PRIOR WRITTEN CONSENT OF A DULY AUTHORIZED REPRESENTATIVE OF HEARTLAND PAYMENT SYSTEMS. IN ORDER TO PROTECT THE CONFIDENTIAL NATURE OF THIS PROPRIETARY INFORMATION, RECIPIENT AGREES:

> *A. TO IMPOSE IN WRITING SIMILAR OBLIGATIONS OF CONFIDENTIALITY AND NON- DISCLOSURE AS CONTAINED HEREIN ON RECIPIENT'S EMPLOYEES AND AUTHORIZED THIRD PARTIES TO WHOM RECIPIENT DISCLOSES THIS INFORMATION (SUCH DISCLOSURE TO BE MADE ON A STRICTLY NEED-TO-KNOW BASIS) PRIOR TO SHARING THIS DOCUMENT AND*

> *B. TO BE RESPONSIBLE FOR ANY BREACH OF CONFIDENTIALITY BY THOSE EMPLOYEES AND THIRD PARTIES TO WHOM RECIPIENT DISCLOSES THIS INFORMATION.*

RECIPIENT ACKNOWLEDGES AND AGREES THAT USE OF THE INFORMATION CONTAINED HEREIN SIGNIFIES ACKNOWLEDGEMENT AND ACCEPTANCE OF THESE TERMS. ANY SUCH USE IS CONDITIONED UPON THE TERMS, CONDITIONS AND OBLIGATIONS CONTAINED WITHIN THIS NOTICE.

THE TRADEMARKS AND SERVICE MARKS RELATING TO PRODUCTS OR SERVICES OF HEARTLAND PAYMENT SYSTEMS OR OF THIRD PARTIES ARE OWNED BY HEARTLAND PAYMENT SYSTEMS OR THE RESPECTIVE THIRD PARTY OWNERS OF THOSE MARKS, AS THE CASE MAY BE, AND NO LICENSE WITH RESPECT TO ANY SUCH MARK IS EITHER GRANTED OR IMPLIED.

TO VERIFY EXISTING CONTENT OR TO OBTAIN ADDITIONAL INFORMATION, PLEASE CALL OR EMAIL YOUR ASSIGNED HEARTLAND PAYMENT SYSTEMS CONTACT.

1. Table of Contents

2. Overview

2.1. Introduction

Katana is a simple HTTP (RESTful) based API for performing basic gift and loyalty transactions against the Heartland Stored Value Platform. It consumes and produces either JSON or XML depending on the clients preference. It also comes with a pre-built Java client that completely hides the transport mechanisms from the client developer.

Most commonly gift and loyalty functionality is provided to a terminal or POS in a restaurant or store. The terminal is configured with credentials that allow authorization of transactions. But end consumer applications such as web sites and mobile apps cannot be coded with credentials to our systems since they are in no way protected or private. Katana was created to provide this functionality to the end consumer. Therefore there are a number of constraints built into the API to prevent fraud and abuse of the API.

To access the Katana API a web site or mobile app developer must apply for an API Key. The API Key is sent by the client with every transaction to Katana. It identifies the application being developed; for example a single web site or a single mobile app. This API Key allows us to identify and manage the clients connecting to our platform. Beyond that the application just needs to know about a few configuration parameters to execute transactions. The configuration parameters are called 'domain' and 'chain' and they are used to identify the Gift Card or Loyalty program that the transactions are being performed against.

Every transaction on this platform also needs to be recorded against a store pre-configured in our system. For example when we do a credit card payment the funds are taken from the credit card and placed in this stores bank. But we can't allow random clients doing transactions by directly specifying a store. Therefore on the server side we configure the system to apply transactions to a store that is configured for the domain, chain and API Key of the client.

The Katana API is a RESTful API. There are three different 'resources' you can access and modify: 'user', 'account' and 'authtoken'. A further resource, 'locations', is read only.

- Users represent the end consumer, they store their username (an email address), password and optionally address information, etc.
- Accounts are repositories of funds which may be a tendered currency such as USD or CAD. They can also contain loyalty currencies such as Points.
- Authtokens are tokens generated by Katana and emailed to users to allow them to validate their email address or change their password.
- Locations are geocoded store locations, that is, the latitude and longitude of stores, for display on maps.

2.2. Audience

This document is provided for software developers building end-consumer applications that will integrate with the Heartland Stored Value Platform via the Katana API.

The end-consumer is a person in possession of a stored value account. Examples of stored value accounts may be gift cards, rewards cards, loyalty cards, prepaid cards, or combinations thereof. In addition, stored value accounts may be physical or virtual. Common examples of physical stored value accounts are plastic gift or loyalty cards, typically with a magnetic stripe on the back. Examples of virtual stored value accounts are accounts stored within a mobile app or digital wallet, or that have been purchased from an e-commerce website.

Katana is designed for consumer-facing websites and mobile apps, with which they can interact with their stored value accounts. It is not intended for merchant-facing websites or mobile apps, and it is not intended for terminal or Point of Sale applications. For these other types of applications, a Heartland specialist can recommend the appropriate API for you.

2.3. Usage / Usecases

Much of the Katana API is self explanatory. This is a description of how you might use the Katana API to accomplish more complicated usecases.

2.3.1. New User

Sequence	Action	API (Method + Endpoint)	Description
1	New User	POST /users	Consumer supplies an email address and password to sign up.
2	Consumer validates email	N/A	Consumer receives email with token and enters it back into your app.
3	Validate Email	POST /authtoken/validate	This makes the user object permanent and marks it as validated. The user can now log in with their email and password and be recognized.

4	Create Account	POST /user/accounts	Create a new account registered to the user with no value.

2.3.2. Forgot Password

Sequence	Action	API (Method + Endpoint)	Description
1	Initiate Forgot Password	N/A	User clicks a link on your website/app to start the process / of getting a new password.
2	Reset Password	GET /users/{user}/authtoken	Causes Katana to invoke your callback with a temporary password for you to email the user.
3	User Returns w/ Token	N/A	Users returns to your website/app with the token and a new password.
4	Set New Password	PUT /authtoken/password	Authenticates the user with the temporary password and changes it to the newly specified password.

2.3.3. Stored Credit Cards

Sequence	Action	API (Method + Endpoint)	Description
1	Add Credit Card	POST /user/creditcards	Saves credit card info to the users profile.
2	Defined Auto-Reload Rule	POST /user/accounts/{account}/autoreloads	Account will now be automatically reloaded when the criteria of the rule is met.
2	Reload Account	POST /user/accounts/{account}/storedload	Reloads the account immediately using the stored credit card.

3. Katana Environments

Katana provides two environments: one for development and test, and the other for the processing of live production transactions.

Environment	URL	Description
Test	https://api.test.chockstone.com/katana	An environment for test and development
Production	https://api.chockstone.net/katana	The environment for live, production use

4. Client Authorization

All clients using the Katana API must include a Heartland-supplied API Key in every request, using the HTTP header "Api-Key". An example API Key is provided below.

```
Api-Key: 8yEsbyXBsaU
```

4.1. Client Authorization Examples

We have provided examples of how to use some common HTTP client libraries to send an API Key in an HTTP header.

4.1.1. Curl

```
curl -H "Api-Key: 8yEsbyXBsaU" https://api.chockstone.net/katana/v1/domains/hps/chains/93729/user
```

4.1.2. C#

```
HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Get,
"https://api.chockstone.net/katana/v1/domains/hps/chains/93729/user");
request.Headers.Add("Api-Key", "8yEsbyXBsaU");
```

4.1.3. Java

```
import org.apache.http.message.BasicHeader;
import org.apache.http.client.methods.HttpGet;

HttpGet request = new HttpGet("https://api.chockstone.net/katana/v1/domains/hps/chains/93729/user");
request.addHeader(new BasicHeader("Api-Key", "8yEsbyXBsaU");
```

4.1.4. Javascript jQuery

```
$.ajaxSetup({
    headers:{ 'Api-Key': '8yEsbyXBsaU' }
});
```

4.1.5. PHP Curl

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPHEADER, array(
    'Api-Key: 8yEsbyXBsaU'
    ));
```

5. Encoding Formats

Katana requires that both the HTTP Content-Type Header and the HTTP Accept Header be sent on all requests, controlling the input and output format of Katana respectivley. At this time, 2 formats are supported:

- `application/json`
- `application/xml` (See also: Katana XSD).

6. Case-Sensitivity

The Katana API is case sensitive! Code samples are provided throughout this document to specify currencies, request bodies, etc. If you do not match the cases exactly, the corresponding features will not work as intended.

7. Authentication

Katana supports an authentication scheme implemented using the standard HTTP Basic Access Authentication protocol. This authentication scheme should be supported by nearly all HTTP client libraries. Because all Katana communication occurs over HTTPS (SSL), the authentication credentials are protected from eavesdropping and are never exposed as plain text.

With very few exceptions, Katana expects clients to provide the appropriate authentication credentials with every request. Details about passing authentication information to Katana using the Basic Access Authentication Protocol can be found here: http://en.wikipedia.org/wiki/Basic_access_authentication

A quick summary for manually constructing an Authorization HTTP header:

1. A username and password are combined into a string "username:password"
2. The resulting string literal is then encoded using Base64
3. Construct the header as follows (where "dXNlcm5hbWU6cGFzc3dvcmQK" is a sample Base64-encoded string from step 2):

```
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQK
```

Manual construction of the Basic Authentication header is typically not required if using a standard HTTP client library for communicating with the Katana host. For example, the following code snippet shows how to pass the authentication credentials using the command line tool curl:

```
curl -u username:password ...
```

7.1. Authentication Types

Katana endpoints are authenticated differently depending on the resource being managed.

7.1.1. None

There are a small number of Katana endpoints that require no authentication at all. In these cases, do not provide any authentication credentials.

Endpoints requiring no authentication begin with the following URIs:

```
/katana/v1/domains/{domain}/chains/{chain}/users
/katana/v1/domains/{domain}/chains/{chain}/locations
```

7.1.2. User Authentication

Most Katana endpoints authenticate a user. This is typically performed using an email address (as the user's username) and password, although support for additional federated identity systems will be added over time. In all cases, there will be an identity passed as the username and a token or phrase passed as the password.

Endpoints requiring User authentication begin with the following URI:

```
/katana/v1/domains/{domain}/chains/{chain}/user
```

7.1.2.1. Email and Password

Username	`babou@e-hps.com`
Password	`5EA7unt`

Example using curl:

```
curl -u babou@e-hps.com:5EA7unt ...
```

7.1.3. Pin Authentication

Some endpoints are designed to operate on a stored value account without first requiring the account to be registered to a user. Instead of a user identifier and password, the account identifier and PIN are used.

Endpoints requiring account PIN authentication begin with the following URI:

```
/katana/v1/domains/{domain}/chains/{chain}/account
```

Username	504422001234567890
Password	123456

Example using curl:

```
curl -u 504422001234567890:123456 ...
```

7.1.4. Authtoken Authentication

Occasionally, such as when creating new users or in support of changing forgotten passwords, a temporary authtoken is sent to the user in an email or other out-of-band message channel. Those authtokens are then used as the password along with the email address as the username.

Endpoints requiring Authtoken authentication begin with the following URI:

```
/katana/v1/domains/{domain}/chains/{chain}/authtoken
```

Username	babou@e-hps.com
Password	yp2Lt61v51

Example using curl:

```
curl -u babou@e-hps.com:yp2Lt61v51 ...
```

8. Errors

Katana uses standard HTTP response codes to indicate success or failure of an API request. In general, 200 means success, 4XX means an application-level error, and 5XX means a system-level error. In addition to the HTTP response code, the response body will contain detailed error information.

An example error response body is provided in JSON below:

```
{
    "code":401,
    "name":"Unauthorized",
    "description":"Missing authorization"
}
```

8.1. HTTP Errors

Standard HTTP errors, see HTTP spec for others.

Status Code	Status Name
400	BadRequest
401	Unauthorized
403	Forbidden
404	NotFound
500	InternalServerError
503	ServiceUnavailable

8.2. Application Errors

Specific errors as translated from the Heartland Stored Value Platform:

Status Code	Status Name
403	ProfileAuthorizationFailed
403	ProfileClosed
403	ProfileFrozen
403	ProfileNotFound

All other errors translated as HTTP 400 (Bad Request).

9. API Summary

9.1. Endpoint Summary

A table is provided below that provides a summary of all Katana endpoints. All endpoints begin with a base URI of: `/katana/v1/domains/domain/chains/chain`.

Therefore, a URI in the table below of `/user/accounts` is translated to: `/katana/v1/domains/domain/chains/chain/user/accounts`.

URI	Auth	Method	Description
`/users`	None	`POST`	Creates unvalidated user and triggers the validation callback
`/users/{user}/forgotpassword`	None	`GET`	Triggers an authtoken callback to change password
`/users/{user}/authtoken`	None	`GET`	Triggers an authtoken callback to validate email
`/authtoken/validate`	Authtoken	`POST`	Validates a user with the authtoken
`/authtoken/password`	Authtoken	`PUT`	Changes a user's password
`/user`	User	`GET`	Gets a user
`/user`	User	`PUT`	Updates a user
`/user`	User	`DELETE`	Deletes a user
`/user/id`	User	`PUT`	Triggers an authtoken callback to change the id/email
`/user/password`	User	`PUT`	Updates a user password
`/user/accounts`	User	`GET`	Gets a user's registered accounts
`/user/accounts`	User	`POST`	Creates a new account and registers it
`/user/accounts/{account}`	User	`PUT`	Updates an account by registering it to the user
`/user/accounts/{account}`	User	`DELETE`	Closes and removes an account if the balances are zero
`/user/accounts/{account}/history`	User	`GET`	Gets account history (activity)
`/user/accounts/{account}/load`	User	`POST`	Loads an account using a credit card
`/user/accounts/{account}/storedload`	User	`POST`	Loads an account using a stored credit card
`/user/accounts/{account}/autoreloads`	User	`GET`	Gets the auto-reload rule(s) for an account
`/user/accounts/{account}/autoreloads`	User	`POST`	Adds a new auto-reload rule for an account
`/user/accounts/{account}/autoreloads/{autoreload}`	User	`DELETE`	Deletes an auto-reload rule for an account
`/user/accounts/{account}/autoreloads/{autoreload}`	User	`PUT`	Modifies an auto-reload rule for an account
`/user/accounts/{account}/pin`	User	`PUT`	Changes an account's pin
`/user/accounts/{account}/aliases`	User	`GET`	Gets the list of aliases for an account
`/user/accounts/{account}/aliases`	User	`POST`	Adds an alias to an account
`/user/accounts/{account}/aliases/{alias}`	User	`DELETE`	Removes an alias from an account
`/user/accounts/{account}/passbook`	User	`GET`	Gets an instance of an account as a Passbook pass
`/user/accounts/transfer`	User	`POST`	Transfers value between two accounts

/user/creditcards	User	GET	Gets a user's stored credit cards
/user/creditcards	User	POST	Adds a stored credit card
/user/creditcards/{creditcard}	User	DELETE	Deletes a stored credit card
/user/optins	User	GET	Get all Opt-Ins in all groups
/user/optins	User	PUT	Update all Opt-Ins in all groups
/user/optins/{group}	User	GET	Get Opt-Ins within a group
/user/optins/{group}	User	PUT	Update Opt-Ins within a group
/user/optins/{group}/{optin}	User	GET	Get a single Opt-In
/user/optins/{group}/{optin}	User	PUT	Update a single Opt-In
/account	Pin	GET	Gets details about an account
/account/history	Pin	GET	Gets account history (activity)
/account/passbook	Pin	GET	Gets an instance of an account as a Passbook pass
/locations	None	GET	Gets a list of store locations, sorted by distance

10. API Reference

10.1. User Management

10.1.1. Get User

Description	Gets a user's own contact info
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user
Authentication	User

10.1.1.1. Request Body

```
N/A
```

10.1.1.2. Response Body

```
{
   "email":"rabbert.klein@e-hps.com",
   "password":null,
   "sms":"5032221212",
   "voice":"5032221212",
   "firstName":"Rabbert",
   "lastName":"Klein",
   "birthDate":"1985-07-31T00:00:00Z",
   "address":{
      "street":"111 Rue des Pyramides",
      "city":"London",
      "state":"OR",
      "zip":"97035",
      "country":"US"
   }
}
```

10.1.2. Create User

Description	Creates a new user and sends a temporary authtoken to the user's email address
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/users
Authentication	None

10.1.2.1. Request Body

Parameter	Required	Type	Description
email	Yes	String	User's email address (will be verified)
password	Yes	String	User's desired password. Must be at least 6 characters.
sms	No	String	User's SMS capable phone number (cell phone)
voice	No	String	User's phone number (potentially same as above)

firstName	No	String	
lastName	No	String	
birthDate	No	DateTime	
address.street	No	String	
address.city	No	String	
address.state	No	String	
address.zip	No	String	
address.country	No	String	

```
{
    "email":"rabbert.klein@e-hps.com",
    "password":"5EA7unt",
    "sms":"5032221212",
    "voice":"5032221212",
    "firstName":"Rabbert",
    "lastName":"Klein",
    "birthDate":"1985-07-31",
    "address":
        {
         "street":"111 Rue des Pyramides",
         "city":"London",
         "state":"OR",
         "zip":"97035",
         "country":"US"
        }
}
```

10.1.2.2. Response Body

```
N/A
```

10.1.3. Validate User

Description	Validate user
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/authtoken/validate
Authentication	Authtoken

10.1.3.1. Request Body

```
N/A
```

10.1.3.2. Response Body

```
N/A
```

10.1.4. Send Temporary Authtoken

Description	Triggers an email to be sent to the user to validate their email address
Method	GET

Request Path	/katana/v1/domains/*domain*/chains/*chain*/users/*user*/authtoken
Authentication	None

10.1.4.1. Request Body

```
N/A
```

10.1.4.2. Response Body

```
N/A
```

10.1.5. Forgot Password

Description	Triggers an email to be sent to the user to help reset a password
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/users/*user*/forgotpassword
Authentication	None

10.1.5.1. Request Body

```
N/A
```

10.1.5.2. Response Body

```
N/A
```

10.1.6. Update User

Description	Overwrite user information. All fields that are not specified will be deleted (except email and password).
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user
Authentication	User

10.1.6.1. Request Body

Parameter	Required	Type	Description
sms	No	String	User's SMS capable phone number (cell phone)
voice	No	String	User's phone number (potentially same as above)
firstName	No	String	
lastName	No	String	
birthDate	No	DateTime	
address.street	No	String	
address.city	No	String	
address.state	No	String	

address.zip	No	String	
address.country	No	String	

```
{
    "sms":"5035551212",
    "voice":"5035551212",
    "firstName":"Rabbert",
    "lastName":"Klein",
    "birthDate":"1985-01-01T00:00:00.000+0000",
    "address":
        {
            "street":"326 SW Broadway",
            "city":"Portland",
            "state":"OR",
            "zip":"97205",
            "country":"US"
        }
}
```

10.1.6.2. Response Body

```
N/A
```

10.1.7. Update User ID

Description	Triggers an email to be sent to the user to confirm the id/email change.
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/id
Authentication	User

10.1.7.1. Request Body

```
{
    "email":"rabby.klein@e-hps.com"
}
```

10.1.7.2. Response Body

```
N/A
```

10.1.8. Update User Password

Description	Change a user's password when the old password is known.
Method	PUT

Authentication	User
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/password

Authentication	Authtoken authentication
Request Path	/katana/v1/domains/*domain*/chains/*chain*/authtoken/password

10.1.8.1. Request Body

Parameter	Required	Type	Description
password	Yes	String	User's desired password. Must be at least 6 characters.

```
{
    "password":"i6QXqr7"
}
```

10.1.9. Delete User

Description	Delete a user from the system. The user's data will no longer be accessible.
Method	DELETE

Authentication	User
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user

10.1.9.1. Request Body

```
N/A
```

10.1.9.2. Response Body

```
N/A
```

10.2. User-Account Registrations

10.2.1. Get Accounts

Description	List accounts registered to user
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts
Authentication	User

10.2.1.1. Request Body

```
N/A
```

10.2.1.2. Response Body

```
{
   "accounts":[
      {
         "id":"120102840000003298",
         "number":"6277209900105081",
         "chainName":"Super laundry",
         "status":"ACTIVE",
         "balances":[
            {
               "currency":"Points",
               "amount":50,
               "formatted":"50"
            },
            {
               "currency":"USD",
               "amount":500,
               "formatted":"5.00"
            }
         ],
         "track2":";6277209900105081=380100012345KATANA?"
      },
      {
         "id":"120102840000003208",
         "number":"6277209900105073",
         "chainName":"Super laundry",
         "status":"ACTIVE",
         "balances":[
            {
               "currency":"Points",
               "amount":50,
               "formatted":"50"
            },
            {
               "currency":"USD",
               "amount":1000,
               "formatted":"10.00"
            }
         ],
         "track2":";6277209900105073=380100012345KATANA?"
      }
   ]
}
```

10.2.2. Register Account

Description	Register an account to a user
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*
Authentication	User

10.2.2.1. Request Body

Parameter	Required	Type	Description
pin	Yes	String	The account PIN
chainId	See [1]	String	Chain in which account will be created.

1. chainId is required when creating accounts for global users (when request path is /katana/v1/domains/*domain*/chains/global/user/accounts).

```
{
    "pin":"54775807",
    "chainId":"290103"
}
```

10.2.2.2. Response Body

```
N/A
```

10.3. Credit Card Management

10.3.1. Get Stored Credit Cards

Description	Gets a user's stored credit cards
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/creditcards
Authentication	User

10.3.1.1. Request Body

```
N/A
```

10.3.1.2. Response Body

```
{
    "storedCreditCards":[
         {"id":"22221504", "lastFour":"2222", expiration:"04/15", "type":"VISA"},
         {"id":"33331605", "lastFour":"3333", expiration:"06/16", "type":"MASTERCARD"},
         {"id":"44441705", "lastFour":"4444", expiration:"04/17", "type":"DISCOVER"}
    ]
}
```

10.3.2. Store Credit Card

Description	Store a credit card
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/creditcards
Authentication	User

10.3.2.1. Request Body

Parameter	Required	Type	Description
number	Yes	String	Credit card number
expiration	Yes	Expiration	Credit card expiration in MM/YY format
cvv2	No	String	Credit card security code
billing.cardholder	Yes	String	Name as it appears on credit card
billing.address.street	No	String	
billing.address.state	No	String	
billing.address.city	No	String	
billing.address.zip	No	String	
billing.address.country	No	String	

```
{
    "number":"4222222222222",
    "expiration":"04/15",
    "cvv2":"123",
    "billing":{
        "cardholder":"Algernop Krieger",
        "address":{
            "street":"326 SW Broadway #400",
            "city":"Portland",
            "state":"OR",
            "zip":"97205-3736",
            "country":"US"
        }
    }
}
```

10.3.2.2. Response Body

```
{
   "id":"44440515",
   "lastFour":"4444",
   "expiration":"05/15",
   "type":"MASTERCARD"
}
```

10.3.3. Delete Stored Credit Card

Description	Delete a stored credit card
Method	DELETE
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/creditcards/*creditcard*
Authentication	User

10.3.3.1. Request Body

```
N/A
```

10.3.3.2. Response Body

```
N/A
```

10.4. Opt-In Management

10.4.1. Get All Opt-Ins

Description	Get Opt-In values across all groups
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins
Authentication	User

10.4.1.1. Request Body

```
N/A
```

10.4.1.2. Response Body

```
{
    "optIns":
    [
        {
            "id": "1001",
            "chainName": "Super laundry",
            "optIns":
            [
                {
                    "id": "EMAILOPTIN",
                    "description": "Opt in to email updates and promotions.",
                    "value": false
                },
                {
                    "id": "SMSMARKETINGOPTIN",
                    "description": "Opt in to promotions via SMS (text message).",
                    "value": false
                },
                {
                    "id": "SMSRECEIPTOPTIN",
                    "description": "Opt in to receipt messages via SMS (text message).",
                    "value": false
                }
            ]
        },
        {
            "id": "1002",
            "chainName": "Suds and duds",
            "optIns":
            [
                {
                    "id": "EMAILOPTIN",
                    "description": "Opt in to email updates and promotions.",
                    "value": false
                },
                {
                    "id": "SMSMARKETINGOPTIN",
                    "description": "Opt in to promotions via SMS (text message).",
                    "value": false
                },
                {
                    "id": "SMSRECEIPTOPTIN",
                    "description": "Opt in to receipt messages via SMS (text message).",
                    "value": false
                }
            ]
        }
    ]
}
```


10.4.2. Update All Opt-Ins

Description	Update Opt-In values across multiple groups
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins
Authentication	User

10.4.2.1. Request Body

Parameter	Required	Type	Description
id	Yes	String	Opt-In group ID

optIns.id	Yes	Opt-In	Opt-In ID
optIns.value	Yes	boolean	Opt-In value

Groups and id/value pairs not sent will remain unaffected; you are not requred to send pairs or group-lists you do not wish to update.

```
{
    "optIns":
    [
        {
            "id": "1001",
            "optIns":
            [
                {
                    "id": "EMAILOPTIN",
                    "value": true
                },
                {
                    "id": "SMSMARKETINGOPTIN",
                    "value": false
                },
                {
                    "id": "SMSRECEIPTOPTIN",
                    "value": true
                }
            ]
        },
        {
            "id": "1002",
            "optIns":
            [
                {
                    "id": "EMAILOPTIN",
                    "value": true
                },
                {
                    "id": "SMSMARKETINGOPTIN",
                    "value": true
                },
                {
                    "id": "SMSRECEIPTOPTIN",
                    "value": true
                }
            ]
        }
    ]
}
```

10.4.3. Get Opt-Ins

Description	Get a group of Opt-In values
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins/*group*
Authentication	User

10.4.3.1. Request Body

```
N/A
```

10.4.3.2. Response Body

```
{
    "id": "1001",
    "chainName": "Super laundry",
    "optIns":
    [
        {
            "id": "EMAILOPTIN",
            "description": "Opt in to email updates and promotions.",
            "value": false
        },
        {
            "id": "SMSMARKETINGOPTIN",
            "description": "Opt in to promotions via SMS (text message).",
            "value": false
        },
        {
            "id": "SMSRECEIPTOPTIN",
            "description": "Opt in to receipt messages via SMS (text message).",
            "value": false
        }
    ]
}
```

10.4.4. Update Opt-Ins

Description	Update a group of Opt-In values
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins/*group*
Authentication	User

10.4.4.1. Request Body

Parameter	Required	Type	Description
id	Yes	Opt-In	Opt-In ID
value	Yes	boolean	Opt-In value

id/value pairs not listed will remain unaffected; you are not required to send an id/value pair for Opt-Ins you do not wish to update.

```
{
    "optIns":
    [
        {
            "id": "EMAILOPTIN",
            "value": true
        },
        {
            "id": "SMSMARKETINGOPTIN",
            "value": false
        },
        {
            "id": "SMSRECEIPTOPTIN",
            "value": true
        }
    ]
}
```

10.4.4.2. Response Body

```
N/A
```

10.4.5. Get Opt-In

Description	Get a single Opt-In value
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins/*group*/*optin*
Authentication	User

10.4.5.1. Request Body

```
N/A
```

10.4.5.2. Response Body

```
{
    "id": "EMAILOPTIN",
    "description": "Opt in to email updates and promotions.",
    "value": false
}
```

10.4.6. Update Opt-In

Description	Update a single Opt-In value
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/optins/*group*/*optin*
Authentication	User

10.4.6.1. Request Body

Parameter	Required	Type	Description
value	Yes	boolean	Opt-In value

```
{
    "value": true
}
```

10.4.6.2. Response Body

```
N/A
```

10.5. Account Management

10.5.1. Create Account

Description	Create a new account
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts
Authentication	User

10.5.1.1. Request Body

Parameter	Required	Type	Description
chainId	See [1]	String	Chain in which account will be created.

1. chainId is required when creating accounts for global users (when request path is /katana/v1/domains/*domain*/chains/global/user/accounts).

```
{
    "chainId":"290103"
}
```

10.5.1.2. Response Body

```
{
    "id":"120102840000003298",
    "number":"5022440100000000118",
    "chainName":"Super laundry",
    "pin":"54775807",
    "status":"Active"
    "track2":";5022440100000000118=380100033158KATANA?"
}
```

10.5.2. Get Account

Description	Get an account
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/account
Authentication	PIN

10.5.2.1. Request Body

```
N/A
```

10.5.2.2. Response Body

```
{
   "id":"120102840000003298",
   "number":"6277209900105008",
   "chainName":"Super laundry",
   "status":"ACTIVE",
   "balances":[
      {
         "currency":"USD",
         "amount":1000,
         "formatted":"10.00"
      }
   ],
   "track2":";6277209900105008=380100011111KATANA?"
}
```

10.5.3. Update Account Pin

Description	Update an account's pin
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/pin
Authentication	User

10.5.3.1. Request Body

Parameter	Required	Type	Description
pin	Yes	String	A new account PIN

```
{
    "pin":"69035124"
}
```

10.5.3.2. Response Body

```
N/A
```

10.5.4. Close Account

Description	Close an account
Method	DELETE
Request Path	/katana/v1/domains/*domain*/chains/*chain*/accounts/*account*
Authentication	User

10.5.4.1. Request Body

```
N/A
```

10.5.4.2. Response Body

```
N/A
```

10.6. Apple Passbook

10.6.1. Get Account Pass

Description	Get a Passbook Pass for an account
Method	GET

Request Path	/katana/v1/domains/*domain*/chains/*chain*/account/passbook
Authentication	PIN

Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/passbook
Authentication	User

10.6.1.1. Request Body

```
N/A
```

10.6.1.2. Response Body

Response below shown with headers; data is a binary .pkpass file.

```
Last-Modified: Mon, 29 Apr 2013 19:05:33 GMT
Date: Thu, 09 May 2013 18:39:29 GMT
Accept-Ranges: bytes
Server: chockstone
Content-Type: application/vnd.apple.pkpass
Content-Disposition: filename="6277200800000001.pkpass"
Content-Length: 196985

[data not shown]
```

10.7. Account History

10.7.1. Get Account History

Description	Get an account's transaction history
Method	GET

Request Path	/katana/v1/domains/*domain*/chains/*chain*/account/history
Authentication	PIN

Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/history
Authentication	User

10.7.1.1. Query parameters

Parameter name	Default value	Description
limit	25	Maximum number of results returned.
page	0	Results page to return; 0 would return history elements 1-25, 1 would return elements 26-50, etc.

order	descending	Chronological sorting direction; "ascending" (oldest first) or "descending" (newest first).

10.7.1.2. Request Body

```
N/A
```

10.7.1.3. Response Body


```
{
   "history":[
      {
         "id":"120102820000020942",
         "type":"Register",
         "date":"2013-07-19T18:20:06Z",
         "store":null,
         "transactionAmount":null,
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
      },
      {
         "id":"120102820000020943",
         "type":"Load",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":1000,
            "formatted":"10.00"
         },
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
      },
      {
         "id":"120102890000000740",
         "type":"Load",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":9502,
            "formatted":"95.02"
         },
         "status":{
            "code":19,
            "name":"CardAuthorizationFailed",
            "description":"Authorization failed: result message [CVV mismatch], response code [0],
result code [0], approval number [015782], authorization number [015782], card number
[XXXXXXXXXXXX0014]."
         }
      },
      {
         "id":"120102820000020946",
         "type":"Redeem",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":-1000,
```

```
            "formatted":"-10.00"
         },
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
      },
      {
         "id":"120102820000020946",
         "type":"Promotion",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":100,
            "formatted":"1.00"
         },
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
      },
      {
         "id":"120102820000020950",
         "type":"Redeem Reversal",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":1000,
            "formatted":"10.00"
         },
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
      },
      {
         "id":"120102820000020950",
         "type":"Promo Reversal",
         "date":"2013-07-19T11:20:39-07:00",
         "store":"Walt's Tasty Store 001",
         "transactionAmount":{
            "currency":"USD",
            "amount":-100,
            "formatted":"-1.00"
         },
         "status":{
            "code":0,
            "name":"okay",
            "description":"okay"
         }
```



```
      }
   ]
}
```

10.8. Account Load

10.8.1. Transfer funds

Description	Transfer account balance
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/transfer
Authentication	User

10.8.1.1. Request Body

Parameter	Required	Type	Description
from.account	Yes	String	Account ID to transfer from
from.pin	See [1]	String	The pin for the account being transferred from, if not registered
to.account	Yes	String	Account ID to transfer to
to.pin	See [1]	String	The pin for the account being transferred to, if not registered

1. PIN is required for any account not registered to the current user.

```
{
   "from":{
      "account":"120102890000002409",
      "pin":"54775807"
   },
   "to":{
      "account":"120102890000002420",
      "pin":null
   }
}
```

10.8.1.2. Response Body

```
{
   "id":"120102890000002420",
   "number":"5032250000000000112",
   "chainName":"1001",
   "status":"ACTIVE",
   "balances":[
      {
         "currency":"Points",
         "amount":10,
         "formatted":"10"
      },
      {
         "currency":"USD",
         "amount":4578,
         "formatted":"45.78"
      }
   ],
   "track2":";5032250000000000112=380100066893KATANA?"
}
```

10.8.2. Stored Credit Card Load

Description	Load an account using a previously stored credit card
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/storedload
Authentication	User

10.8.2.1. Request Body

Parameter	Required	Type	Description
loadAmount.amount	Yes	Amount	An integer amount to load
loadAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.)
storedCreditCardId	Yes	String	ID of stored credit card

```
{
    "loadAmount":
    {
        "amount":350,
        "currency":"USD"
    },
    "storedCreditCardId":"22221504"
}
```

10.8.2.2. Response Body

```
{
    "order":"120102620000000284",
    "authorizedAmount":
    {
        "amount":350,
        "currency":"USD",
        "formatted":"3.50"
    },
    "paymentType":"VISA",
    "authorizationNumber":"20090A",
    "notes":"Thanks for visiting!",
    "balances": [
        { "currency":"USD",    "amount":1558, "formatted":"15.58" },
        { "currency":"Points", "amount":542,  "formatted":"542" }
    ],
    "rewards": [
        { "currency":"Points", "amount":10, "formatted":"10" }
    ]
}
```

10.8.3. Credit Card Load

Description	Load an account using a credit card
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/load
Authentication	User

10.8.3.1. Request Body

Parameter	Required	Type	Description
loadAmount.amount	Yes	Amount	An integer amount to load
loadAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.)
number	Yes	String	Credit card number
expiration	Yes	Expiration	Credit card expiration in MM/YY format
cvv2	Yes	String	Credit card security code
billing.cardholder	Yes	String	Name as it appears on credit card
billing.address.street	No	String	
billing.address.state	No	String	
billing.address.city	No	String	
billing.address.zip	No	String	
billing.address.country	No	String	

```
{
    "loadAmount":
    {
        "amount":1301,
        "currency":"USD"
    },
    "number":"4222222222222",
    "expiration":"04/15",
    "cvv2":"123",

    "billing":{
        "cardholder":"Algernop Krieger",
        "address":{
            "street":"326 SW Broadway #400",
            "state":"OR",
            "city":"Portland",
            "zip":"97205-3736",
            "country":"US"
        }
    }
}
```

10.8.3.2. Response Body

```
{
    "order":"120102620000000284",
    "authorizedAmount":
    {
        "amount":1301,
        "currency":"USD",
        "formatted":"13.01"
    },
    "paymentType":"VISA",
    "authorizationNumber":"20090A",
    "notes":"Thanks for visiting!",
    "balances": [
        { "currency":"USD",    "amount":1558, "formatted":"15.58" },
        { "currency":"Points", "amount":542,  "formatted":"542" }
    ],
    "rewards": [
        { "currency":"Points", "amount":10, "formatted":"10" }
    ]
}
```

10.9. Automatic Reload Management

10.9.1. Get Auto-Reload Rules

Description	Gets the auto-reload rule(s) for an account
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/autoreloads
Authentication	User

10.9.1.1. Request Body

```
N/A
```

10.9.1.2. Response Body

```
{
    "autoReloads":
    [
        {
            "start":"2013-07-10T00:00:00Z",
            "end":"2013-07-10T00:00:00Z",
            "thresholdAmount":
            {
                "currency":"USD",
                "amount":1000,
                "formatted":"10.00"
            },
            "dayOfMonth":0,
            "storedCreditCardId":"22220422",
            "reloadAmount":
            {
                "currency":"USD",
                "amount":1000,
                "formatted":"10.00"
            },
            "id":"120100200000004773"
        }
    ]
}
```

10.9.2. Create Auto-Reload Rule

Description	Setup automatic account reloading using a stored credit card
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/autoreloads
Authentication	User

10.9.2.1. Request Body

Parameter	Required	Type	Description
start	Yes	DateTime	Date to start automatic account loading
end	Yes	DateTime	Date to stop automatic account loading

thresholdAmount.amount	See [1]	Amount	Perform automatic load if account balance falls below this amount.
thresholdAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.) - must match reloadAmount.currency below.
dayOfMonth	See [1]	Number	Perform automatic load on this day of the month.
storedCreditCardId	Yes	String	ID of stored credit card
reloadAmount.amount	Yes	Amount	An integer amount to load
reloadAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.)

1. The 'thresholdAmount' and 'dayOfMonth' parameters are not mutually exclusive. If either parameter is non-zero then it will trigger an automatic reload only when both conditions are met.

```
{
    "start":"2013-05-18T07:00:00Z",
    "end":"2014-05-18T07:00:00Z",

    "thresholdAmount":
    {
        "amount":100,
        "currency":"USD"
    },
    "dayOfMonth":0,

    "storedCreditCardId":"22221504",
    "reloadAmount":
    {
        "amount":20000,
        "currency":"USD"
    }
}
```

10.9.2.2. Response Body

```
{
    "id":"5022440100000000118"
}
```

10.9.3. Delete Auto-Reload Rule

Description	Disable an automatic account reload rule
Method	DELETE
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/autoreloads/*autoreload*
Authentication	User

10.9.3.1. Request Body

```
N/A
```

10.9.3.2. Response Body

```
N/A
```

10.9.4. Modify Auto-Reload Rule

Description	Modify a rule performing automatic account reloading
Method	PUT
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/autoreloads/*autoreload*
Authentication	User

10.9.4.1. Request Body

Parameter	Required	Type	Description
start	Yes	DateTime	Date to start automatic account loading
end	Yes	DateTime	Date to stop automatic account loading
thresholdAmount.amount	See [1]	Amount	Perform automatic load if account balance falls below this amount.
thresholdAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.) - must match reloadAmount.currency below.
dayOfMonth	See [1]	Number	Perform automatic load on this day of the month.
storedCreditCardId	Yes	String	ID of stored credit card
reloadAmount.amount	Yes	Amount	An integer amount to load
reloadAmount.currency	Yes	Currency	ISO 4217 currency code (USD, CAD, etc.)

1. The 'thresholdAmount' and 'dayOfMonth' parameters are not mutually exclusive. If either parameter is non-zero then it can trigger an automatic reload when conditions are met.

```
{
    "start":"2013-05-18T07:00:00Z",
    "end":"2014-05-18T07:00:00Z",

    "thresholdAmount":
    {
        "amount":0,
        "currency":"USD"
    },
    "dayOfMonth":16,

    "storedCreditCardId":"22221504",
    "reloadAmount":
    {
        "amount":20000,
        "currency":"USD"
    }
}
```

10.9.4.2. Response Body

```
N/A
```

10.10. Account Alias Management

10.10.1. Get Account Aliases

Description	Gets the list of aliases for an account
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/aliases

Authentication	User

10.10.1.1. Request Body

```
N/A
```

10.10.1.2. Response Body

```
"aliases":[
       {"id":"5035551212"},
       {"id":"5035551313"}
    ]
```

10.10.2. Add Account Alias

Description	Adds an alias to an account
Method	POST
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/aliases
Authentication	User

10.10.2.1. Request Body

```
{
    "id":"5035551212"
}
```

10.10.2.2. Response Body

```
N/A
```

10.10.3. Remove Account Alias

Description	Removes an alias from an account
Method	DELETE
Request Path	/katana/v1/domains/*domain*/chains/*chain*/user/accounts/*account*/aliases/*alias*
Authentication	User

10.10.3.1. Request Body

```
N/A
```

10.10.3.2. Response Body

```
N/A
```

10.11. Store Locations

Description	Retrieves a list of locations for all stores within a chain, sorted by distance to the mobile device making the request. Results are returned as a latitude/longitude pair, and a distance in miles.
Method	GET
Request Path	/katana/v1/domains/*domain*/chains/*chain*/locations
Authentication	None

10.11.1.1. Request Body

```
N/A
```

10.11.1.2. Response Body

```
{
   "storeLocations":
   [
      {"name":"Pizzeria 9","chainName":"Fantastic
Pizza","location":{"lat":45.51369010000001,"lng":-122.6643746},"distance":3.25752260},
      {"name":"Pizzeria 2","chainName":"Fantastic
Pizza","location":{"lat":45.5193129,"lng":-122.6790466},"distance":4.053371348785007},
      {"name":"Pizzeria 6","chainName":"Fantastic
Pizza","location":{"lat":43.3917071,"lng":-124.2717642},"distance":167.3808620490151},
      {"name":"Pizzeria 1","chainName":"Fantastic
Pizza","location":{"lat":37.42919680000001,"lng":-121.9056287},"distance":558.790674},
      {"name":"Pizzeria 5","chainName":"Fantastic
Pizza","location":{"lat":32.7153292,"lng":-117.1572551},"distance":929.7053985252049}
   ]
}
```

11. Appendix A - Parameter Reference

11.1. Expiration

Description	Credit card expiration date in MM/YY format.
Type	String
Accepted input values	Valid MM/YY string (MM: `01-12`).
Examples	`11/15` `02/14`
Notes	

11.2. Tender amount

Description	Identifies the amount of a financial transaction. This parameter always expects an implied decimal point for the associated currency. For example, `1000` for the USD currency has an implied decimal point as if it were rendered as `10.00` representing ten dollars USD. The location of the implied decimal point is currency-specific.
Type	Integer
Accepted input values	Positive integer (`1-2147483647`)
Examples	`1000`
Notes	Amounts returned by Katana may be negative (e.x.: "Redeem" transaction amount in account history).

11.3. Tender currency

Description	Identifies the currency of a financial transaction, using an ISO 4217 currency code.
Type	String
Accepted input values	Valid ISO 4217 string.

Examples	`USD`
	`CAD`
	`NZD`

Notes	Katana returns the non-ISO currency "Points" for loyalty rewards. Points cannot be loaded directly, and as such all endpoints *accept* ISO 4217 only.

11.4. Date

Description	Standard ISO 8061 DateTime format, can be used with JavaScript Date.toISOString() or Date.getTime() (milliseconds since epoch). ISO strings that do not include a time or timezone will be interpreted in UTC.
Type	DateTime
Accepted input values	Valid ISO 8061 Date/DateTime, or time in milliseconds (64 bit).

Examples	`2014-05-18T07:00:00Z`
	`2013-07-19T11:20:39-07:00`
	`2001-10-26`
	`1374602085423`

Notes	Katana will always return a full ISO 8061 string with both time and timezone.

11.5. Opt-In

Description	Identifies an Opt-In program (E-Mail, SMS, etc.) a user can participate in.
Type	Enumerated set of String values, see Opt-In IDs.
Accepted input values	See Opt-In IDs.
Examples	`EMAILOPTIN` `SMSRECEIPTOPTIN`
Notes	

12. Appendix B - Enumerated Values

12.1. Opt-In IDs

The complete set of supported OptIn IDs are provided below and are case sensitive.

Value	Description
`EMAILOPTIN`	Opt in to email updates and promotions.
`SMSMARKETINGOPTIN`	Opt in to promotions via SMS (text message).
`SMSRECEIPTOPTIN`	Opt in to receipt messages via SMS (text message).