Legacy Push Notification Capabilities

Cheetah Messaging

Last Modified: November 2021

Version History

Version	Date	Description	Reviewed / Approved by
1.0	March 2018	Initial release	Cheetah Digital Product Management
2.0	November 2019	Minor updates	Cheetah Digital Product Management
2.1	November 2021	Review, added version history	Cheetah Digital Product Management

Table of Contents

1 Introduction 4

Overview 4

Push vs. Texting 4

2 Process Flow 6

Messaging Set-Up 7

Audience Segmentation and Personalization 8

Segmentation 8

Personalization 8

Integration Options 9

Direct Mobile App vs Server-to-Server Integration 9

Software Development Kit 9

Launching 9

Mobile App Reporting and Analytics 10

Standard Reporting 10

Analytics and Events 10

Third Party Integrations 10

3 Templates 11

Overview 11

Push Notification Template Screen 11

Push Notification Campaign Screen 12

JSON Payload Fields 13

Standard Required Elements 14

iOS Example 14

Android Example 14

Custom Elements 15

Android Example 16

iOS Example 16

1 Introduction

Overview

The purpose of this document is to provide a high-level overview of the Push Notification capabilities offered by the Cheetah Messaging platform.

A Push Notification is a message that's sent to a specific application on a customer's mobile device. With Messaging, you can create and manage marketing Campaigns with Push Notifications, allowing you to engage with your customers by sending messages that are personalized, relevant, and tightly integrated with your mobile app.

Push Notifications are enabled when a customer installs your app on his or her mobile device, and agrees to receive notifications. On Android devices, users must grant permission to receive Push Notifications in order to install the app. On iOS devices, users will be prompted after the app is installed. If the customer opts-in, then he or she is a viable candidate for your marketing Campaigns. Unlike email or SMS text messages, if customers opt-out of Push Notifications on their device, they won't receive Push Notifications, even if they are sent.

If the associated app isn't running in the foreground on the customer's device, then the device's operating system will display an alert, notifying the user that a new notification has been received for this app. Typically, this alert displays only a portion of the entire notification. The user must then open the app to view the entire contents of your Push Notification.

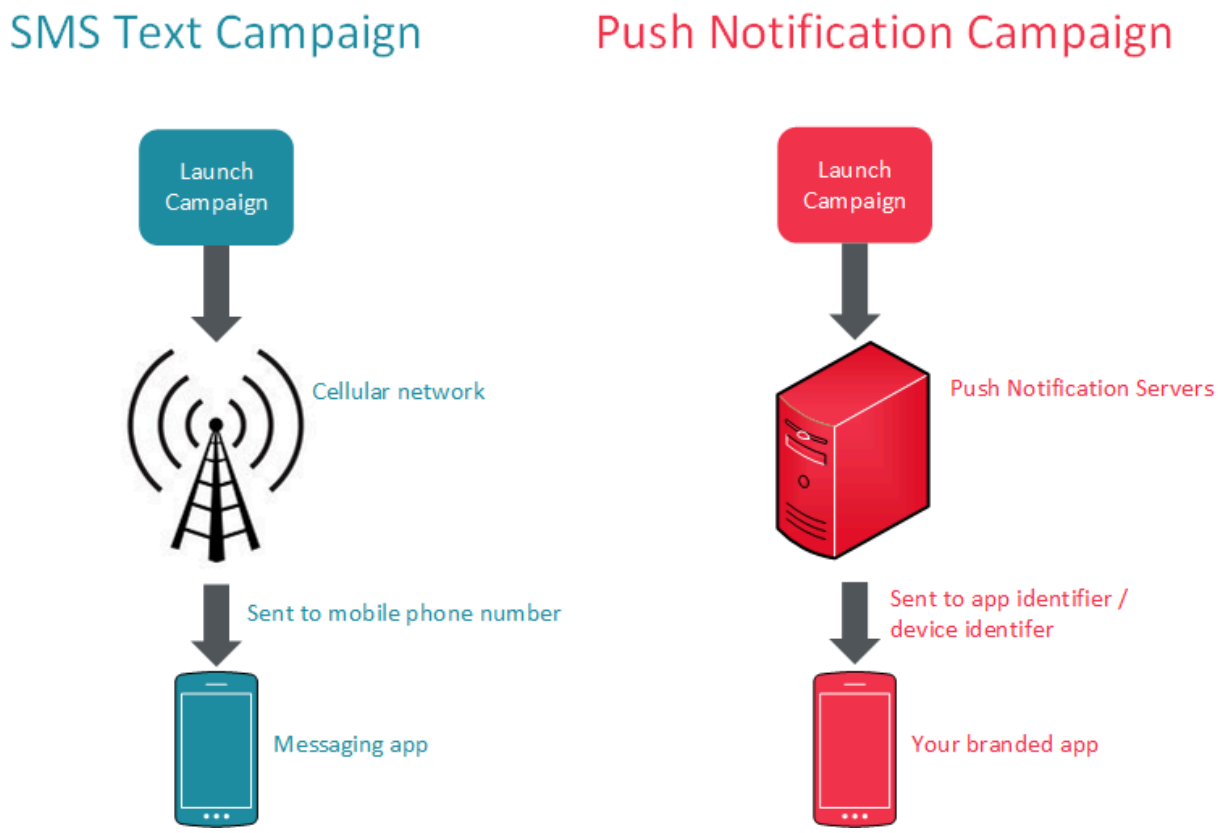
Push vs. Texting

Texting requires a mobile phone number in order to send a text message to the customer. Push Notifications, on the other hand, are sent over the internet using the device

platform's Push Notification Servers. Instead of using a cell phone number as the recipient identifier of your customers, Push Notifications use a device token. Therefore, users can potentially receive Push Notifications without providing any Personally Identifiable Information (PII).

Since text messages are sent over the cellular phone network, they fall within the bounds of the customer's cellular service contract. For customers who don't have an "unlimited texting" option with their service provider, your marketing messages sent via SMS texting count against their text message limit. Push Notifications can count towards a user's data plan if the user is not on a local wireless network when receiving the notifications.

SMS texting uses communication features and apps built directly into most smartphones, with no development effort on your part. Push Notifications, on the other hand, require users to install your app from an app store. However, by delivering timely and relevant marketing messages with your app, customers will be enticed to open and engage with your app.



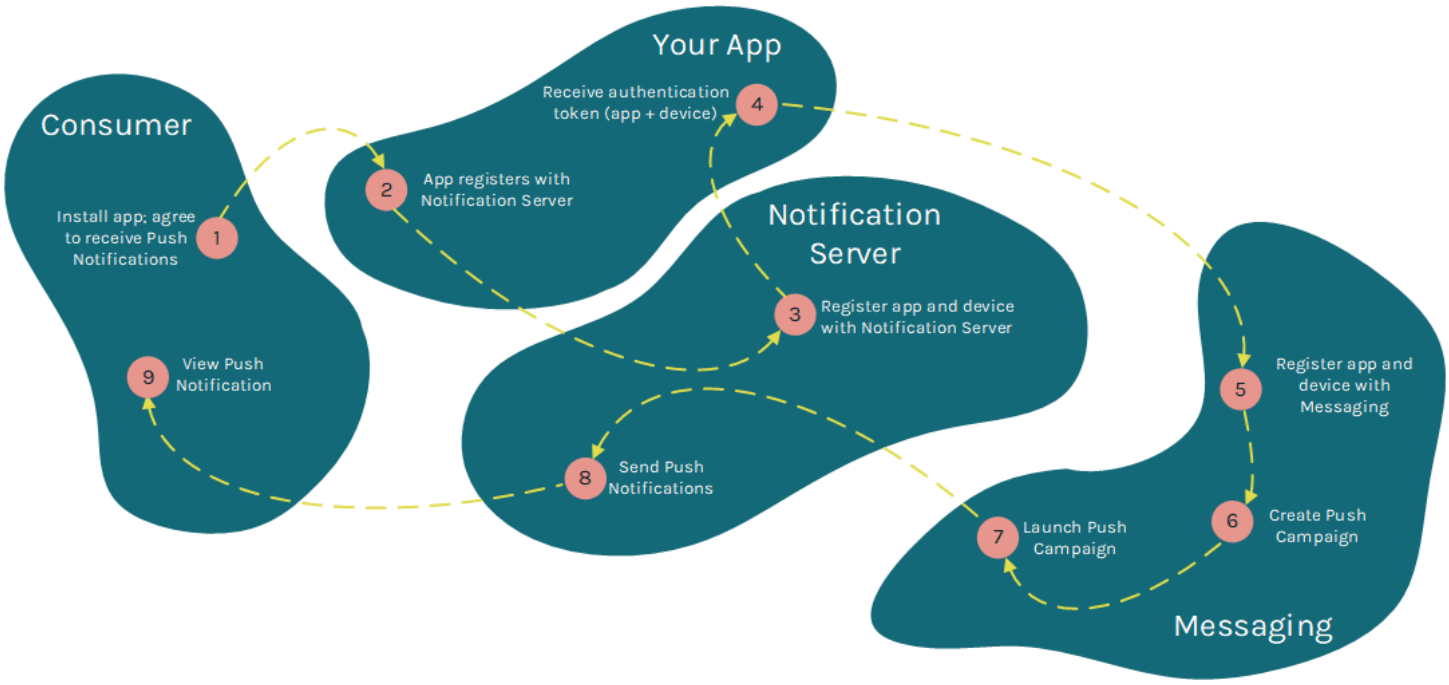
2 Process Flow

Push Notifications require communication between three separate entities:

1. Your app on the customer's mobile device.
2. The platform Notification Servers (iOS-Apple, Android-Google).
3. Cheetah Messaging platform.

The basic flow for Push Notifications is to gather the necessary identifying information (app + device) from customers' mobile devices, then to use that information to build the audience for your Campaign.

The following diagram depicts this process flow.



The first step in the Push Notification process is for the customer to download and install your app on his or her device, and agree to receive Push Notifications. The app must then

register with the platform's Push Notification Server, and receive back a unique "device token." This token is necessary to send a Push Notification to an app on a device.

The app must then send this device token along with a customer ID and application ID to Messaging. The platform uses this token to generate a new unique identifier called a "Push Registration ID" or "PRID." The PRID is used within Messaging for campaign marketing purposes to link together the individual, the device, and the app.

At this point, you're ready to build your Campaign audience, define the message content, and launch your Campaign. As with any marketing channel in Messaging, your Push Notification Campaign can be a one-time broadcast, or it can be configured to be date-triggered or event-triggered. The audience can be segmented, and the content of the message can be personalized.

When your Campaign is launched, Messaging sends the message to the platform Notification Servers, along with the "device token." The Notification Servers then send your message out to the targeted customers.

Messaging Set-Up

To use the Push Notification channel in Messaging, you must go through some one-time configuration steps.

1. **Create a Push Notification Template** -- The Push Notification Template screen provides a JSON editor that can be used to define a JSON form schema. This form schema is a set of instructions that determines the appearance and options that are available when building the content of a Push Notification Campaign. Essentially, you're creating a custom data entry form that's displayed on the Campaign screen. See below for more details on the Push Notification Template.
2. **Configure your app in Messaging** -- You can organize your apps into "Application Groups," which allows you to launch a single Campaign that gets deployed to all apps within the Application Group. An Application Group contains the credentials required by each platform (iOS/Android) to send Push Notifications. Within Messaging, the Application Group serves much the same purpose as a Sender Profile in an Email Campaign, in that it controls the recipient's eligibility to be contacted. You can also

assign one or more Push Notifications Templates to the apps within the Application Group.

3. **Create the Push Notification table** -- The information needed for Push Notification marketing must be stored in a separate table that will serve as the source table for your marketing Campaign. The PRID field in this table must then be linked to at least one Application Group.
4. **Map device registration information** -- A data map is used to associate the Push Notification table with the mobile app group. When Messaging receives a new device registration, the PRID and platform name are added to the Push Notification table.

Audience Segmentation and Personalization

When a Push Registration ID becomes associated with a customer ID, such as when a user logs in on their mobile app, Messaging can then make the association between the anonymous PRID and a customer's personal information. This association allows Push Notification Campaigns to be targeted to a specific audience, and the message content to be personalized.

Segmentation

Similar to other Campaign channels, such as email and SMS text, Push Campaign audiences can be filtered based upon demographics and/or user activity across channels. For example, you could send a Push Notification only to the audience that didn't open your last email message.

Personalization

The Push Notification message body and metadata (JSON) payload can be personalized with content based upon the user's demographic or engagement history. Simple personalization, such as name and location, can be accomplished through merge tags. More advanced, dynamic content can be used to provide completely customized message content. For example, if you're sending a Push Notification about a one-day footwear sale, you could include specific text and a link to women's shoes for your female customers, and different text and link to men's shoes for your male customers.

Integration Options

Direct Mobile App vs Server-to-Server Integration

Direct app integration involves your app communicating with Messaging either through APIs or by using the Mobile Software Development Kit (SDK). Push registration and data exchange occur directly between the mobile app and Messaging. Alternatively, your app can communicate with your backend server, and that server can handle all of the integration with Messaging. Each approach has its advantages, and they are not mutually exclusive. Your developers and existing implementations will determine the best options.

Software Development Kit

The Messaging Mobile Software Development Kit (SDK) is a file containing a library of functions that make it faster and easier for your mobile app developers to integrate your app directly with Messaging. Your mobile app developers can download the SDK and incorporate it into your app.

Currently, the SDK provides the following functions:

- Register the mobile application with Messaging to receive Push Notifications.
- Send event data to Messaging. Event data can be anything the mobile app developers wish to report to Messaging.

Launching

A Push Notification Campaign launches in a similar manner to other channels. In the case of Push Notifications, a mobile app is required for a mobile device to receive the notification. If the app is in the background or not running, the “Alert” message of the Push Notification will be displayed on the mobile device’s home or lock screen. If the user taps on the Push Notification, the associated mobile app will open and the app can process the JSON payload data to link the user to a specific page within the app or to display personalized information.

Mobile App Reporting and Analytics

Standard Reporting

The standard Push Campaign Report indicates the total number of sent messages and app opens. An app open event is recorded when a user swipes a push notification, which will open the associated application.

Analytics and Events

Any user activity data collected within the mobile app, such as last page viewed, button clicks, and abandoned cart, can be transferred to the Messaging through the API or Mobile SDK. This data can be stored in multiple joined tables on the platform. This mobile data, as well as user activity collected in other channels (such as email opens, for example), can be used in Filters for targeting, as well as for reports and dashboards.

Third Party Integrations

Third party analytics platforms can be used to transfer relevant mobile activity data to Messaging through the API, or through recurring batch file transfers from the analytics platform to Messaging. These data imports can then be used for segmentation and reporting.

Conversely, data from Messaging can be sent to third parties as API calls through the Webhook feature, or through scheduled file exports.

3 Templates

Overview

Push Notifications are delivered to mobile devices as JSON -- a human readable text data format. Clients can fully customize and configure the JSON that gets sent as part of their Push Notifications.

In addition to the “Alert” section of a Push Notification (what customers see pop up on their mobile device’s home or lock screen), other sections of the JSON can be used for a variety of data that can be used by the mobile application to perform specific actions, or to display personalized information within the app. For example, clients can configure the Push Notification to include a “deep link” to open the app to a specific page within the app, to play a video, to show order information, or to facilitate campaign attribution.

The JSON editor is implemented in two areas of the Messaging platform:

1. **Push Notification Template** – This screen allows clients to define the JSON form schema. The schema is a set of instructions that define the editable form that will appear in the "Push Payload" section of the Campaign screen.
2. **Push Notification Campaign Screen** -- The Push payload form is displayed when configuring the Push Notification Campaign, allowing the user to select or enter the desired parameter values.

Push Notification Template Screen

Within the Push Notification Template screen, users can configure a Push payload form using a JSON schema. The JSON editor is, in part, based on an open-source tool. More information about how to use a JSON form schema can be found here: https://github.com/jdorn/json-editor.

The JSON form schema determines the appearance and options appearing in the Push payload form. Field names, default values, as well as enumerated values that will appear in the form as pull-down menus, can be defined here. For example, you can optionally assign parameters as:

- **Hidden** -- Hidden fields aren't displayed on the Push Notification Campaign screen.
- **Read-only** -- Read-only fields are displayed on the Push Notification Campaign screen, but aren't editable by the end-user.
- **Advanced** -- Advanced fields aren't displayed by default on the Push Notification Campaign screen; they optionally can be displayed if the user chooses them from an "Advanced Options" menu.

Like any web form, the fields in the schema can be configured as simple text entry fields or as drop-down menus in order to limit user choices and to avoid typos. The structure of the custom JSON will need to be determined by the needs of the marketers and the design of the mobile app. While the setup is the same for all Campaigns based on the selected Template, the values can be modified for each Campaign.

To use a Push Notification Template, you must first assign it to one or more apps within an Application Group. When this Application Group is selected as the Sender Profile for the Campaign, the user will be able to select the desired Template.

Push Notification Campaign Screen

To create Push Notification Campaigns in Messaging, you can use either the legacy "classic" Campaign screen, or you can use the Enhanced Workflow Campaign screen. This document focuses on the Enhanced Workflow Campaign screen for building Push Notification Campaigns.

The Enhanced Workflow version of the Campaign screen is part of a larger initiative designed to provide enhancements around creating and launching Campaigns in Messaging. The Enhanced Workflow is available in parallel with the existing "classic" Campaign screen. Any changes made through the Enhanced Workflow will be applied and

saved to your Campaign. Use of the Enhanced Workflow is entirely optional, and you can continue to utilize the familiar "classic" Campaign screen if you prefer.

The key new features available in the Enhanced Workflow Campaign screen are as follows:

- A cleaner, more streamlined user interface.
- Usability improvements to the Personalization Pane where you select Personalization Fields and other assets.

When an Application Group is selected for a Campaign, the user must then select the desired Template, from a list of all Templates assigned to an app within the Application Group. The Push Notification Campaign screen then displays the JSON editor form defined for the selected Template.

If the Template has settings for both iOS and Android, you'll see options to edit the payload for both operating systems together, or you can toggle to edit each operating system individually. If the selected Template has settings for only one platform, you'll see just the editable options for that platform.

Please note that parameters designated as "hidden" won't be displayed in the "Push Payload" section. Also, any parameters designated as "read only" will be displayed, but you can't edit them.

If the selected Template has parameters that were designated as "Advanced," then an "Advanced Options" menu is displayed within the "Push Payload" section. To view an Advanced parameter, select it from the "Advanced Options" menu.

JSON Payload Fields

Both iOS and Android devices receive JSON payloads, but the two platforms differ from one another in how they handle those payloads.

iOS devices are stricter in enforcing which parts of the JSON payload are used for displaying the contents of the Push Notification on the screen.

For Android devices, the developer has more flexibility in determining from which section of the JSON payload the subject and body of the notification are derived.

Standard Required Elements

In their simplest form, below are examples of the Push Notification JSON received by both iOS and Android.

iOS Example

```
{
  "aps": {
    "alert": "This is the body.",
    "badge": 0,
    "content-available": 0,
    "sound": ""
  },
  "ems_open":
"http:\/\/localhost\/rts\/open.aspx?tp=i-H8B-1c-7qu-7Yb3-Xv-D4I-1c-GW-
7Yb1-1CnUFi",
  "ems_campaign_id": "30188",
  "ems_message_id": "1801289",
  "ems_send_time": "2017-01-14T01:21:29.2869103Z”
}
```

The “aps” section contains the **alert** parameter, which is the body of the message appearing on the screen. All other parameters under the “aps” section (**badge, content-available**, and **sound**) adhere to functionality provided by Apple, and can alter the appearance and functionality of the application when a notification is received.

Android Example

```
{
  "ems_open":
"http:\/\/localhost\/rts\/open.aspx?tp=i-H8B-1c-7qu-7Yb2-Xv-D4H-1c-GW-
7Yb1-15C5CN",
  "ems_campaign_id": "30188",
  "ems_message_id": "1801288",
  "ems_send_time": "2017-01-14T01:12:42.2609234Z",
  "body": "This is the body.",
  "title": "This is the subject"
}
```

By default, the Android JSON payload contains the **body** and **title** parameters used to display the content of the Push Notification on the Android device.

Both iOS and Android share some common parameters, which are added automatically by Messaging. The most important of these keys is the **ems_open** key. This key is a URL which allows tracking of receipt of the message by the recipient. Based on behavior by the

recipient (e.g. swiping the Push Notification to open the application), the URL can be accessed in the background via HTTP GET. Once this action is performed, it's added to the tally of messages for a specific Campaign which were opened. Subsequently, Filters and reports can be generated using the "Open" criteria on a per-Campaign basis.

Custom Elements

In the case of both iOS and Android, the JSON payload allows the developer to add JSON structures to the Push Notification payload. The structure and content of this section is entirely determined by the mobile app developers, because it passes data only to the mobile application, rather than to the iOS or Android operating systems. The custom section is not required.

The custom section can be used to pass a variety of data to the mobile app including:

- A deep link to open the app to a specific screen, such as the user's shopping cart.
- Personalized data that can be processed by the mobile app to display specific order status, offers, or relevant products.
- Contextual information based on the user's location, such as the nearest store or local events.

Below is an example of a **custom_data** entry, which demonstrates some of the JSON types available in a JSON payload (string, array, and object):

```
{
  "platform": {
    "ios": {
      "AppleNotificationPayload": {
        "Badge": 0,
        "Category": "",
        "ContentAvailable": 0,
        "HideActionButton": false,
        "Sound": "",
        "AppleNotificationAlert": {
          "ActionLocalizedKey": "",
          "LaunchImage": "",
          "LocalizedArgs": [

          ],
          "LocalizedKey": ""
        }
      }
    },
    "gcm": {
```

```
      "dry_run": false
    }
  },
  "custom_data": {
    "simple_key1": "simple_value1",
    "array1": [
      "array1_index0",
      "array1_index1"
    ],
    "object1": {
      "object1_key1": "object1_key1_val1",
      "object1_nested1": {
        "object1_nested1_key1": "object1_nested1_value1"
      }
    }
  }
}
```

Android Example

```
{
  "ems_open":
"http:\/\/localhost\/rts\/open.aspx?tp=i-H8B-1c-87P-7bBs-Xv-D4H-1c-GW-
7bBr-2IsHEy",
  "ems_campaign_id": "31211",
  "ems_message_id": "1811260",
  "ems_send_time": "2017-01-23T18:22:18.3683426Z",
  "simple_key1": "simple_value1",
  "array1": [
    "array1_index0",
    "array1_index1"
  ],
  "object1": {
    "object1_key1": "object1_key1_val1",
    "object1_nested1": {
      "object1_nested1_key1": "object1_nested1_value1"
    }
  },
  "body": "This is a standard body",
  "title": "This is a standard subject"
}
```

iOS Example

```
{
  "aps": {
    "alert": "This is a standard body",
    "badge": 0,
    "content-available": 0,
    "sound": ""
  },
  "ems_open":
"http:\/\/localhost\/rts\/open.aspx?tp=i-H8B-1c-87P-7bBt-Xv-D4I-1c-GW-
7bBr-MC6qp",
  "ems_campaign_id": "31211",
```

```
    "ems_message_id": "1811261",
    "ems_send_time": "2017-01-23T18:24:21.4951086Z",
    "simple_key1": "simple_value1",
    "array1": [
      "array1_index0",
      "array1_index1"
    ],
    "object1": {
      "object1_key1": "object1_key1_val1",
      "object1_nested1": {
        "object1_nested1_key1": "object1_nested1_value1"
      }
    }
  }
```

In both cases above, all entries in the **custom_data** section are siblings to one another, as well as to the **ems_ prefixed** keys. In the case of iOS, there is a requisite **aps** object, which contains the message to be displayed on the lock screen of the device.