# LiveEngage Enterprise In-App Messenger SDK Deployment Guide: iOS v2.1.5


Quick Start

- Prerequisites
- Step 1: Download and unzip the SDK
- Step 2: Configure project settings to connect LiveEngage SDK
- Step 3: Initialization

Advanced Configurations

- SDK Initialization and Lifecycle
- SDK Delegates
- The SDK uses 2 delegates:
- Authentication
- UI
- Conversation Lifecycle
- Notifications
- User Data
- Logs and Info
- App Extensions

API Methods

- initialize
- showConversation
- removeConversation


**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- reconnect
- toggleChatActions
- checkActiveConversation
- markAsUrgent
- isUrgent
- dismissUrgent
- resolveConversation
- clearHistory
- logout
- destruct
- handlePush
- registerPushNotifications
- setUserProfile
- getAssignedAgent
- subscribeLogEvents
- getSDKVersion
- printAllLocalizedKeys
- printSupportedLanguages
- getAllSupportedLanguages

Interface and class definitions

- LPUser
- LPLog
- LogLevel
- LPNotification
- LPConversationCloseReason


**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

Callbacks Index

- LPMessagingSDKDelegate
- LPMessagingSDKCustomButtonTapped
  - LPMessagingSDKAgentDetails(agent: LPUser?)
  - LPMessagingSDKAgentAvatarTapped(_ agent: LPUser?)
  - LPMessagingSDKActionsMenuToggled(toggled: Bool)
  - LPMessagingSDKHasConnectionError(error: String?)
  - LPMessagingSDKCSATScoreSubmissionDidFinish(brandID: String, rating: Int)
  - LPMessagingSDKCSATCustomTitleView(_ brandID: String) -> UIView
  - LPMessagingSDKObseleteVersion(error: NSError)
  - LPMessagingSDKAuthenticationFailed(error: NSError)
  - LPMessagingSDKTokenExpired
  - LPMessagingSDKError(error: NSError)
  - LPMessagingSDKAgentIsTypingStateChanged(isTyping: Bool)
  - LPMessagingSDKConversationStarted(conversationID: String?)
  - LPMessagingSDKConversationEnded(conversationID: String?)
  - LPMessagingSDKConversationEnded(_ conversationID: String?, closeReason: LPConversationCloseReason)
  - LPMessagingSDKConversationCSATDismissedOnSubmittion(conversationID: String?)
  - LPMessagingSDKConnectionStateChanged
  - LPMessagingSDKOffHoursStateChanged(isOffHours: Bool, brandID: String)
  - LPMessagingSDKConversationViewControllerDidDismiss()
- LPMessagingSDKNotificationDelegate
  - LPMessagingSDKNotification(didReceivePushNotification notification: LPNotification)
  - LPMessagingSDKNotification(shouldShowPushNotification notification: LPNotification) -> Bool

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

LPMessagingSDKNotification(customLocalPushNotificationView notification: LPNotification) -> UIView

LPMessagingSDKNotification(notificationTapped notification: LPNotification)

Configuring the SDK

Attributes

Users Bubble

Link Preview

Photo Sharing

Send Button

System Messages

Window Mode

Delivery Notifications

CSAT Buttons

CSAT Survey

Conversations

Unread Messages

scrollToBottomButtonBackgroundColor

scrollToBottomButtonMessagePreviewTextColor

scrollToBottomButtonBadgeBackgroundColor

scrollToBottomButtonBadgeTextColor

scrollToBottomButtonArrowColor

unreadMessagesDividerBackgroundColor

unreadMessagesDividerTextColor

scrollToBottomButtonEnabled

scrollToBottomButtonMessagePreviewEnabled

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

unreadMessagesDividerEnabled

Localization

Brand

Date Separator

User input view

Agent Assignment

Duration of Local Notifications

Time To Response and Off hours

Date and Time

Toast Notifications

User Avatar

Data Masking

Navigation

Deprecated Attributes

String localization in SDK

Timestamps Formatting

Time/Date Styles

Off Hours

Bubble timestamp

Separator timestamp

Resolve message

OS Certificate Creation

Creating a key .p12 file

Creating both key.pem file and cert.pem file

CSAT Behavior


**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- Overview
- Show CSAT flow
- Dismiss CSAT
- CSAT UI content
  - Agent View (avatar and agent name)
  - Rating Question View (stars)
  - Resolution Confirmation View (yes/no)

Photo Sharing

- Overview
- Enable Photo Sharing
- Upload Photo

LiveEngage Configuration

- Push Notifications

Open Source List

Appendix

- Security


# Quick Start

The LivePerson SDK provides brands with a simple, yet enterprise-grade and secure in-app messaging solution. Through in-app messaging, brands will foster connections with their customers and increase app engagement and retention.

This Quick Start will quickly get you up and running with a project powered by LivePerson. When you're done, you'll be able to send messages between an iOS device and LiveEngage. To complete this Quick Start, you will need a LiveEngage account. You can get the number and login information from the LivePerson account team.

## Prerequisites

To use the LivePerson In-App Messaging SDK, the following are required:

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- XCode 8.3 or later
- Swift 3.1 or later, or Objective-C

*Note: For information on supported operating systems and devices, refer to System Requirements.*

## Step 1: Download and unzip the SDK

1. Click here to download the SDK package.

2. Once downloaded, extract the ZIP file to a folder on your Mac.

## Step 2: Configure project settings to connect LiveEngage SDK

1. Copy the files into the project.
2. In project settings, navigate to the **General** tab, and add all Framework files to the **Embedded Binaries** section.

3. In the **General** tab, make sure that the framework files are under **Embedded Libraries**.
4. In Build settings, make sure **Always Embed Swift Standard Libraries** is set to **YES**.
5. Due to a new Apple policy for iOS 10 (or later), apps must declare in their project settings which privacy settings may be used. For more information, refer to Apple's website.

   In Xcode info.plist of the project, add two new privacy keys and values:

   - Key: NSPhotoLibraryUsageDescription, Value: "Photo Library Privacy Setting for LiveEngage iOS SDK",
   - Key: NSCameraUsageDescription, Value: "Camera Privacy Setting for LiveEngage iOS SDK"

   This step is required in order to be able to upload your host app into the App Store, as SDK 2.0 has the ability to share photos from the camera and/or photo library.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

*Note: Due to Apple policy, this step is mandatory even if the photo sharing feature is disabled in the SDK.*

6. In project settings, navigate to the **Build Phases** tab, and click the **+** button to add a **New Run Script Phase**. Add the script below in order to loop through the frameworks embedded in the application and remove unused architectures (used for simulator). This step is a workaround for known iOS issue http://www.openradar.me/radar?id=6409498411401216 and is necessary for archiving your app before publishing it to the App Store.

   7.1 If frameworks installed using CocoaPods, use the following script:

```
bash "${SRCROOT}/Pods/LPMessagingSDK/LPMessagingSDK/LPInfra.framework/frameworks-strip.sh"
```

   7.2 If frameworks installed using copy to Xcode project, use the following script:

```
bash "${BUILT_PRODUCTS_DIR}/${FRAMEWORKS_FOLDER_PATH}/LPInfra.framework/frameworks-strip.sh"
```

## Step 3: Initialization

1. Inside **`AppDelegate`**, under **`didFinishLaunchingWithOptions`**, add the following code:

```
do {
   try LPMessagingSDK.instance.initialize("Your Account ID")

} catch {
   // SDK has an initialization error...
   return

}
```

2. Set up and call the conversation view. You'll need to provide your LivePerson account number and a container view controller.

```
let conversationQuery =
   LPMessagingSDK.instance.getConversationBrandQuery(accountNumber)

LPMessagingSDK.instance.showConversation(conversationQuery, authenticationCode:
   accountNumber, containerViewController: self)
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

3. In order to remove the conversation view when your container is deallocated, run the following code:

```
let conversationQuery =
   LPMessagingSDK.instance.getConversationBrandQuery(accountNumber)

LPMessagingSDK.instance.removeConversation(conversationQuery)
```

# Advanced Configurations

## SDK Initialization and Lifecycle

In order to get started and initialize the In-App Messaging SDK, utilize the following functions:

```
1. public func initialize(brandID: String? = nil) throws
2. public func showConversation(conversationQuery: ConversationParamProtocol,
   authenticationCode: String? = nil, containerViewController: UIViewController? = nil)
3. public func removeConversation(conversationQuery: ConversationParamProtocol)
4. <LPMessagingSDKdelegate>   optional func LPMessagingSDKConnectionStateChanged(isReady:
   Bool, brandID: String)
5. <LPMessagingSDKdelegate>   func LPMessagingSDKError(error: NSError)
6. <LPMessagingSDKdelegate>   optional func LPMessagingSDKHasConnectionError(error:
   String?)
```

Supporting functions:

```
1. public func getConversationBrandQuery(brandID: String) -> ConversationParamProtocol
2. public func getConversationBrandAndSkillQuery(brandID: String, skillID: String) ->
   ConversationParamProtocol
3. public func getConversationConsumerQuery(consumerID: String?, brandID: String,
   agentToken: String) -> ConversationParamProtocol
4. public func isBrandReady(brandID: String) -> Bool
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## SDK Delegates

The SDK uses 2 delegates:

1. LPMessagingSDKDelegate - for lifecycle and connectivity events.
2. LPMessagingSDKNotificationDelegate - for handling push and in app notifications.

You should implement and set the delegate for the above in order to receive notification from the SDK.

## Authentication

For users of OAuth 2.0 for customer authentication, the following functions apply:

```
1. public func reconnect(conversationQuery: ConversationParamProtocol, authenticationCode:
   String)
2. <LPMessagingSDKdelegate> func LPMessagingSDKAuthenticationFailed(error: NSError)
3. <LPMessagingSDKdelegate> func LPMessagingSDKTokenExpired(brandID: String)
```

## UI

To determine the layout of messaging within the app, you can utilize various actions to control the behavior and UI such as menus, custom buttons, typing indication, etc.

```
1. public func toggleChatActions(accountID: String, sender: UIBarButtonItem? = nil)
2. <LPMessagingSDKdelegate>   optional func LPMessagingSDKActionsMenuToggled(toggled:
   Bool)
3. <LPMessagingSDKdelegate>   optional func LPMessagingSDKCustomButtonTapped()
4. <LPMessagingSDKdelegate>   optional func
   LPMessagingSDKAgentIsTypingStateChanged(isTyping: Bool)
5. <LPMessagingSDKdelegate>   optional func LPMessagingSDKOffHoursStateChanged(isOffHours:
   Bool, brandID: String)
6. <LPMessagingSDKdelegate>   optional func
   LPMessagingSDKConversationViewControllerDidDismiss()
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## Conversation Lifecycle

During the course of the conversation, consumers can take several actions such as Mark as urgent to receive a faster service, or Resolve conversation to let your agents know they have received their answers.

```
1. <LPMessagingSDKdelegate>   optional func
   LPMessagingSDKConversationStarted(conversationID: String?)
2. <LPMessagingSDKdelegate>   optional func
   LPMessagingSDKConversationEnded(conversationID: String?)
3. func LPMessagingSDKConversationEnded(_ conversationID: String?, closeReason:
   LPConversationCloseReason)
4. public func checkActiveConversation(conversationQuery: ConversationParamProtocol) ->
   Bool
5. public func markAsUrgent(conversationQuery: ConversationParamProtocol)
6. public func isUrgent(conversationQuery: ConversationParamProtocol) -> Bool
7. public func dismissUrgent(conversationQuery: ConversationParamProtocol)
8. public func resolveConversation(conversationQuery: ConversationParamProtocol)
9. <LPMessagingSDKdelegate>  optional func
   LPMessagingSDKConversationCSATDismissedOnSubmittion(conversationID: String?)
10. <LPMessagingSDKdelegate>  optional func
    LPMessagingSDKCSATScoreSubmissionDidFinish(brandID: String, rating: Int)
11. public func clearHistory(conversationQuery: ConversationParamProtocol) throws
12. public func logout()
13. public func destruct()
```

## Notifications

Push and local notifications are a key factor that make the experience better for consumers - they never have to stay in your app or keep the window open as they will get a proactive notification as soon as a reply or notice is available.

*Note: In order to enable push notifications, you must also configure them within the LiveEngage UI.* *See instructions.*

```
1. public func handlePush(userInfo: [NSObject : AnyObject])
2. public func registerPushNotifications(token token: NSData, notificationDelegate:
   LPMessagingSDKNotificationDelegate? = nil, alternateBundleID: String? = nil)
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
3.    <LPMessagingSDKNotificationDelegate> optional func LPMessagingSDKNotification
    (didReceivePushNotification notification: LPNotification)
4.  <LPMessagingSDKNotificationDelegate>  optional func
    LPMessagingSDKNotification(shouldShowPushNotification notification: LPNotification) ->
    Bool
5.  <LPMessagingSDKNotificationDelegate>   optional func
    LPMessagingSDKNotification(customLocalPushNotificationView notification:
    LPNotification) -> UIView
6.  <LPMessagingSDKNotificationDelegate>   optional func
    LPMessagingSDKNotification(notificationTapped notification: LPNotification)
```

## User Data

Pass and display consumer information to agents, and agent information to consumers.

```
1.  public func setUserProfile(lpuser: LPUser, brandID: String)
2.  public func getAssignedAgent(conversationQuery: ConversationParamProtocol) -> LPUser?
3.  <LPMessagingSDKdelegate>  optional func LPMessagingSDKAgentDetails(agent: LPUser?)
```

## Logs and Info

Send logs from LiveEngage to your app. Logs include different severity levels of errors and warnings.

```
1.  public func subscribeLogEvents(logLevel: LogLevel)
2.  <LPMessagingSDKdelegate>  func LPMessagingSDKObseleteVersion(error: NSError)
3.  public func getSDKVersion() -> String?
4.  public func printAllLocalizedKeys()
5.  public func printSupportedLanguages()
6.  public func getAllSupportedLanguages() -> [String : String]
```

## App Extensions

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

In order to make sure the SDK uses the iOS keyboard only, and not third party ones, disable app extensions for keyboard as follows:

In your AppDelegate, add the method `application(_:shouldAllowExtensionPointIdentifier:)`

with the implementation of:

```
func application(_ application: UIApplication, shouldAllowExtensionPointIdentifier
    extensionPointIdentifier: UIApplicationExtensionPointIdentifier) -> Bool {
    return extensionPointIdentifier != UIApplicationExtensionPointIdentifier.keyboard
 }
```

# API Methods

| Name | Description |
|---|---|
| initialize | Initialize the resources required by the SDK. |
| showConversation | Open the conversation screen. |
| removeConversation | Remove the SDK UI and clean the service or network operation that was running. |
| reconnect | Reconnect with new authentication key. |
| toggleChatActions | Open or close the SDK menu. |
| checkActiveConversation | Check whether there is an active conversation. |
| markAsUrgent | Mark the current conversation as urgent. |
| isUgent | Check whether the current conversation is marked as urgent. |
| dismissUrgent | Cancel the markAsUrgent API. |
| resolveConversation | Resolve the current conversation. |
| clearHistory | Clear all conversations from device. |
| logOut | Logout from the SDK - when all user data should be removed. |
| destruct | Close all active connections and remove conversation view. |
| handlePush | Receive all incoming push messages in a single function. |
| registerPushNotifications | Register to LPMessagingSDK push notifications. |
| setUserProfile | Take custom parameters about the consumer as an input, set them for the messaging agent, and attach them to the |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| | transcript. |
| getAssignedAgent | Get assigned agent details of last / current conversation. |
| subscribeLogEvents | Subscribe to log events. |
| getSDKVersion | Get current SDK version string. |
| printAllLocalizedKeys | Prints all localized strings keys |
| printSupportedLanguages | Prints the SDK supported languages |
| getAllSupportedLanguages | Get all supported languages as Strings dictionary |

## initialize

The SDK initialization is done only once, inside **`appdelegate`**. This function checks that the SDK has all mandatory preconditions. For example, it is able to find the bundle file, verify that all the pre-defined configurations are valid, and more. If any of the preconditions are not met, an exception is thrown. Once an exception is thrown, you must not do any other call to the SDK.

| *func initialize(_ brandID: String? = nil) throws* | |
|---|---|
| brandId | An account ID |

## showConversation

This method is used to open the conversation screen.

| *func showConversation(_ conversationQuery: ConversationParamProtocol, authenticationCode: String? = nil, containerViewController: UIViewController? = nil)* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |
| authenticationCode | The SDK can enable code-flow SSO. If your account uses SSO, pass the auth-code here. Otherwise, skip this parameter. |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| containerViewController | The SDK needs a container view controller. This can be done in two ways:<br><br>a. View Controller mode. If you provide a container viewController, the SDK will put itself inside as a child viewController. This mode allows you to keep your own navigation bar intact. Using this method, you can use the provided callbacks to retrieve data from the SDK and show it in the navigation bar (users profile data, avatar URL, calling menu items, etc.)<br><br>b. Window mode<br><br>If you don't provide a container view controller, the SDK places its UI components on top of the app UI, including the navigation bar. |

## removeConversation

When navigating out of the conversation screen, remove the view controller from its container. This is done by calling remove conversation API. The method will remove the SDK UI and clean the service or network operation that was running.

| *func removeConversation(_ conversationQuery: ConversationParamProtocol)* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## reconnect

When using SSO in an authenticated connection, an auth-code is passed to the SDK (see `showConversation` API). The session in this case might have an expiration date (see `LPMessagingSDKTokenExpired`). To reconnect with a new token, use the following 'reconnect' API and pass the new token.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| *func reconnect(_ conversationQuery: ConversationParamProtocol, authenticationCode: String)* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |
| authenticationCode | The SDK can enable code-flow SSO. If your account uses SSO, pass the auth-code here. Otherwise, skip this parameter. |

## toggleChatActions

This API call is used to open or close the SDK menu.

- If you're using window mode, you won't need to utilize this method as the SDK will have a dedicated button in the navigation bar to toggle the menu.
- If you are using view controller mode, you may call this API to open the SDK menu, or use other APIs to build your own menu.

| *func toggleChatActions(_ accountID: String, sender: UIBarButtonItem? = nil)* | |
|---|---|
| *accountID* | An account ID |
| sender | An optional UIBarButtonItem to use for toggling the chat actions |

## checkActiveConversation

Check if there is an active conversation by passing a conversation query.

- Note that conversation query defines a filter that fetches conversations which match certain conditions. Each query can have one active conversation at most.
- Conversation is said to be active the moment an 'ack' is received from the server. It may not yet have an assigned agent.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- You may call this API only if you are sure that the SDK is in sync with the server, meaning LPMessagingSDKConnectionStateChanged was invoked and isReady is set to true.

| *func checkActiveConversation(_ conversationQuery: ConversationParamProtocol) -> Bool* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## markAsUrgent

A consumer can mark a conversation as urgent in order to request a faster response from the agent. You can call the API only if there's an active conversation, otherwise an alert will be triggered. The conversation is marked as urgent only after an ACK is received from the server.

| *func markAsUrgent(_ conversationQuery: ConversationParamProtocol)* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## isUrgent

Checks if the active conversation (if existing) is marked as urgent. Otherwise returns false.

Note that you must check that the SDK is in ready state before calling this method.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| *func isUrgent(_ conversationQuery: ConversationParamProtocol) -> Bool* | |
|---|---|
| conversationQuery | The conversation query represents a ‘filter’ for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## dismissUrgent

This API is used to cancel the markAsUrgent API. It will reset the SLA for the agent response back to default.  This API can be called only for open conversations.

| *func dismissUrgent(_ conversationQuery: ConversationParamProtocol)* | |
|---|---|
| conversationQuery | The conversation query represents a ‘filter’ for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## resolveConversation

This API enables a conversation to be resolved. The API will request the server to mark the active conversation as resolved. If there is no active conversation, an alert will be displayed.

| *func resolveConversation(_ conversationQuery: ConversationParamProtocol)* | |
|---|---|
| conversationQuery | The conversation query represents a ‘filter’ for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number. |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| | See helpers methods above for how to generate a conversation query. |

## clearHistory

This API may be used only when there is no active conversation. This API clears the local database. The history is still available on the server, but won't be retrieved from the specific device unless a fresh installation is made.

| *func clearHistory(_ conversationQuery: ConversationParamProtocol) throws* | |
|---|---|
| conversationQuery | The conversation query represents a 'filter' for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## logout

This method is a destructive method that is typically used to clean a user's data before a second user logs into the same device. This method conducts the following:

- Unregisters from the push notification service.
- Clears all SDK persistent data.
- Cleans running operations (see destruct).

| *func logout()* |
|---|

## destruct

| *func destruct()* |
|---|

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## handlePush

In order to receive all incoming push notifications in a single function and handle them, add the following method. This method cooperates with two other API methods:

- This method calls the shouldShowPushNotification method. If the host app returns false, the SDK will not show anything to the UI.
- Otherwise, the SDK will ask the host app to provide a view as an in-app notification. If the host app doesn't implement this method, the SDK will use its own implementation.

| *func handlePush(_ userInfo: [AnyHashable : Any])* | |
|---|---|
| userInfo | A dictionary that contains information related to the remote notification. This is the exact same dictionary as received in application(_:didReceiveRemoteNotification:fetchCompletionHandler:) method |

## registerPushNotifications

Register to LPMessagingSDK push notifications with the following code in AppDelegate:

- Note that push notifications must be pre-configured, and an APN certificate has to be uploaded to the LiveEngage platform. See more info on how to configure push notifications.

| *func registerPushNotifications(token: Data, notificationDelegate: LPMessagingSDKNotificationDelegate? = nil, alternateBundleID: String? = nil)* | |
|---|---|
| token | A token that identifies the device to APNs. The token is an opaque data type because that is the form that the provider needs to submit to the APNs servers when it sends a notification to a device. The APNs servers require a binary format for performance reasons.<br><br>This is the exact same dictionary as received in |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| | application:didRegisterForRemoteNotificationsWithDeviceToken:<br><br>method |
| notificationDelegate | An implementer of LPMessagingSDKNotificationDelegate. |
| alternateBundleID | An optional value that can be used so that the LivePerson pusher service identifies your app with this identifier.<br><br>In debug mode, the SDK appends “-dev” string to the bundle ID. |

## setUserProfile

Add custom parameters about the user and set them for the messaging agent.

| *func setUserProfile(_ lpuser: LPUser, brandID: String)* | |
|---|---|
| lpuser | object is an instance of LPUser.<br>Example:<br>let user = LPUser(firstName: “John”, lastName: “Doe”, profileImageURL: “URL of image”, phoneNumber: “555-555555”) |
| brandId | An account ID |

## getAssignedAgent

Get assigned agent details of the last or current conversation - depending on retrieveAssignedAgentFromLastClosedConversation in the LPConfig defaultConfiguration.

You must check that the SDK is ready before calling this method.

| *func getAssignedAgent(_ conversationQuery: ConversationParamProtocol) -> LPUser?* | |
|---|---|
| conversationQuery | The conversation query represents a ‘filter’ for the conversation screen, determining which of the conversations will be displayed in the following screens. The default conversationQuery sorts the |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| | conversations by account number.<br><br>See helpers methods above for how to generate a conversation query. |

## subscribeLogEvents

Subscribe to log events (Trace, Debug, Info, Warning, Error). Each time a log event with the passed log level occurs, the callback will be invoked with the log object.

| *func subscribeLogEvents(_ logLevel: LogLevel, logEvent: @escaping LogEventClosure)* | |
|---|---|
| logLevel | object is an instance of LPLog. |
| logEvent | The completion block will pass LPLog object which consists all the information for the log. |

## getSDKVersion

Get current SDK version string.

| *func getSDKVersion() -> String?* |
|---|

## printAllLocalizedKeys

Prints all localized strings' keys

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

*func printAllLocalizedKeys()*

### printSupportedLanguages

Prints the SDK supported languages

*func printSupportedLanguages()*

### getAllSupportedLanguages

Get all supported languages as Strings dictionary where:
Key - Locale key
Value - explicit language name (example: "English", "Portuguese (Brazil)", ...)

*func getAllSupportedLanguages() -> [String : String]*

## Interface and class definitions

### LPUser

```
class LPUser: NSObject {

	var firstName: String?

	var lastName: String?

	var profileImageURL: String?

	var phoneNumber: String?

	var employeeID: String?
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
        var uid: String?

}
```

## LPLog

```
class LPLog: NSObject {

        var timestamp: String?

        var className: String?

        var funcName: String?

        var text: String?

        var logLevel: LogLevel?

}
```

## LogLevel

```
enum LogLevel: Int {

        case TRACE

        case DEBUG

        case INFO

        case WARNING

        case ERROR

        case OFF

}
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## LPNotification

```
class LPNotification: NSObject {

       var text: String

       var user: LPUser

       var accountID: String

       var isRemote: Bool

       var toString : String

}
```

## LPConversationCloseReason

```
public enum LPConversationCloseReason: Int {

     case agent = 0

     case consumer

     case system

}
```

# Callbacks Index

The SDK uses 2 delegates:

1. LPMessagingSDKDelegate - for lifecycle and connectivity events
2. LPMessagingSDKNotificationDelegate - for handling push and in app notifications

## LPMessagingSDKDelegate

```
protocol LPMessagingSDKdelegate {
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
    optional func LPMessagingSDKCustomButtonTapped()

    optional func LPMessagingSDKAgentDetails(_ agent: LPUser?)

    optional func LPMessagingSDKAgentAvatarTapped(_ agent: LPUser?)

    optional func LPMessagingSDKActionsMenuToggled(_ toggled: Bool)

    optional func LPMessagingSDKHasConnectionError(_ error: String?)

    optional func LPMessagingSDKCSATScoreSubmissionDidFinish(_ brandID: String, rating: Int)

    optional func LPMessagingSDKCSATCustomTitleView(_ brandID: String) -> UIView



    func LPMessagingSDKObseleteVersion(_ error: NSError)

    func LPMessagingSDKAuthenticationFailed(_ error: NSError)

    func LPMessagingSDKTokenExpired(_ brandID: String)

    func LPMessagingSDKError(_ error: NSError)



    optional func LPMessagingSDKAgentIsTypingStateChanged(_ isTyping: Bool)

    optional func LPMessagingSDKConversationStarted(_ conversationID: String?)

    optional func LPMessagingSDKConversationEnded(_ conversationID: String?)

    optional func LPMessagingSDKConversationEnded(_ conversationID: String?, closeReason:
LPConversationCloseReason)

    optional func LPMessagingSDKConversationCSATDismissedOnSubmittion(_ conversationID:
String?)

    optional func LPMessagingSDKConnectionStateChanged(_ isReady: Bool, brandID: String)

    optional func LPMessagingSDKOffHoursStateChanged(_ isOffHours: Bool, brandID: String)

    optional func LPMessagingSDKConversationViewControllerDidDismiss()

}
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

### LPMessagingSDKCustomButtonTapped

In window mode only, the app can place a custom button in the SDK UI. When the button is tapped, the following delegate method is invoked:

```
LPMessagingSDK.instance.delegate = self


When this button is pressed, it will call the following delegate:


func LPMessagingSDKCustomButtonTapped() {

    UIApplication.sharedApplication().openURL(NSURL(string: "tel://55555555")!)

}
```

### LPMessagingSDKAgentDetails(agent: LPUser?)

Called when agent details are received or updated. For every message, the SDK checks for agent details in order to determine whether the assigned agent was changed. If there is no assigned agent, agent will be nil, for instance, when the consumer is returned to queue.

### LPMessagingSDKAgentAvatarTapped(_ agent: LPUser?)

Called when the user tapped on the agent's avatar in the conversation and also in the navigation bar within window mode.

### LPMessagingSDKActionsMenuToggled(toggled: Bool)

Called when the action menu is toggled.

### LPMessagingSDKHasConnectionError(error: String?)

Called whenever the SDK receives a connection error from the socket.

### LPMessagingSDKCSATScoreSubmissionDidFinish(brandID: String, rating: Int)

When a conversation is resolved, a feedback page is presented (CSAT - Customer Satisfaction).

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

This delegate method is invoked after the CSAT is submitted. If the user chooses to skip the CSAT, the delegate method is called with score=0.

### LPMessagingSDKCSATCustomTitleView(_ brandID: String) -> UIView

Custom Title view for to display in the CSAT survey view.

### LPMessagingSDKObseleteVersion(error: NSError)

Called when the SDK version is obsolete and needs to be updated.

### LPMessagingSDKAuthenticationFailed(error: NSError)

Called when the current session fails due to an authentication error.

### LPMessagingSDKTokenExpired

Called when the current session fails due to an authentication error.

### LPMessagingSDKError(error: NSError)

Called when the SDK has a general error.

If there is an SDK initialization error, the SDK can not proceed, and you should not call any other SDK API.

There are also other possible errors such as send message error.

### LPMessagingSDKAgentIsTypingStateChanged(isTyping: Bool)

Called when the typing state of the agent is changed.

### LPMessagingSDKConversationStarted(conversationID: String?)

Called when a new conversation is started.

### LPMessagingSDKConversationEnded(conversationID: String?)

Called when an open conversation is ended by the consumer or by the agent or automatically by the system.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

### LPMessagingSDKConversationEnded(_ conversationID: String?, closeReason: LPConversationCloseReason)

Called when an open conversation is ended by the consumer or by the agent or automatically by the system. Close reason is the conversation ended with type LPConversationCloseReason

### LPMessagingSDKConversationCSATDismissedOnSubmittion(conversationID: String?)

Called after CSAT screen is dismissed by clicking **Submit**.

### LPMessagingSDKConnectionStateChanged

Invoked when the connection state is changed.

- isReady: Bool - Set to true when the SDK is connected and in sync with the server.
- brandID: String - Brand account number

### LPMessagingSDKOffHoursStateChanged(isOffHours: Bool, brandID: String)

Delegate which is called when an off hours state changes.

### LPMessagingSDKConversationViewControllerDidDismiss()

Delegate which is called when the conversation viewcontroller is dismissed (both for window mode and viewController mode).

## LPMessagingSDKNotificationDelegate

```
protocol LPMessagingSDKNotificationDelegate {

    optional func LPMessagingSDKNotification(didReceivePushNotification notification:
LPNotification)

    optional func LPMessagingSDKNotification(shouldShowPushNotification notification:
LPNotification) -> Bool

    optional func LPMessagingSDKNotification(customLocalPushNotificationView notification:
LPNotification) -> UIView

    optional func LPMessagingSDKNotification(notificationTapped notification: LPNotification)
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
}
```

### LPMessagingSDKNotification(didReceivePushNotification notification: LPNotification)

Called when handling the push notifications, the SDK will form up a struct: LPNotification,

which will then be passed to the host app.

### LPMessagingSDKNotification(shouldShowPushNotification notification: LPNotification) -> Bool

Called after calling handlePush, the SDK will ask the host app if it should display an in-app notification in the UI. (See handlePush for the full description).

### LPMessagingSDKNotification(customLocalPushNotificationView notification: LPNotification) -> UIView

If shouldShowPushNotification is not implemented, or it returns yes, the app can implement this method for showing an in-app notification in the UI. In case the method is not implemented, the SDK will provide and show its own view. Note that setting the custom view's Frame is advised and not its Bounds.

```
func LPMessagingSDKNotification(customLocalPushNotificationView notification:
LPNotification) -> UIView {
    guard let delegate = UIApplication.shared.delegate as? AppDelegate, let window
=   delegate.window else { return UIView() }

   view.frame = CGRect(x: 0, y: 0, width: window.bounds.size.width, height: 85)
   return view
 }
```

### LPMessagingSDKNotification(notificationTapped notification: LPNotification)

Called when tapping a local notification message bar when a remote push notification received. You should implement this delegate method if you wish to navigate and show the conversation screen.

```
 func LPMessagingSDKNotification(notificationTapped notification: LPNotification) {
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
    // Navigate to a desired view controller

}
```

## Configuring the SDK

The SDK allows you to configure the look and feel of your app using LPConfig object. To apply a custom look and feel, create your own configuration instance and assign the attributes you want to customize.

**The most suitable time to customize configuration is right after the SDK initialization and before calling showConversation().**

To get the default configuration:

```
let configuration = LPConfig.defaultConfiguration
```

To print all configurable attributes and their default values call:

```
LPConfig.printAllConfigurations()
```

To customize an attribute, follow this example:

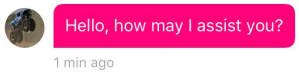
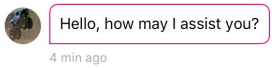
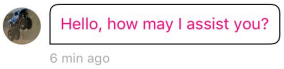
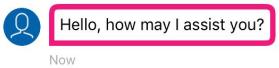
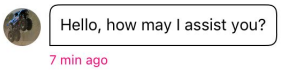
```
configuration.remoteUserBubbleBackgroundColor = UIColor.purpleColor()

configuration.brandName = “Brand Name”

configuration.remoteUserBubbleBorderWidth = 0.5
```

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

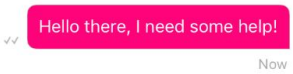
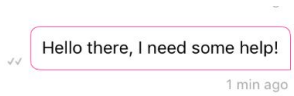
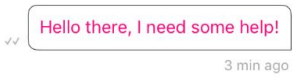
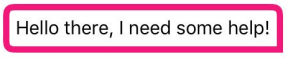
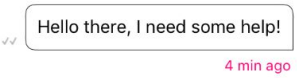
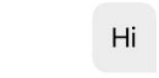
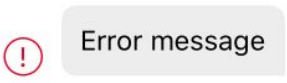
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## Attributes

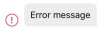
The table below lists the available attributes which can be used to personalize your app. The **Default / Customer notes** column includes space for you to add your own branding.

| Name | Type | Description | Example | Default |
|---|---|---|---|---|
| **Users Bubble** | | | | |
| `remoteUserBubbleBackgroundColor` | `UIColor` | Color code for the background of the remote user bubble. |  | #004DC9 |
| `remoteUserBubbleBorderColor` | `UIColor` | Color code for the outline color. |  | #004DC9 |
| `remoteUserBubbleLinkColor` | `UIColor` | Color code for links in the text of the remote user bubble. | | UIColor.white |
| `remoteUserBubbleTextColor` | `UIColor` | Color code for the text of the remote user bubble. |  | UIColor.white |
| `remoteUserBubbleBorderWidth` | `Double` | Double number for the outline width. |  | 2 |
| `remoteUserBubbleTimestampColor` | `UIColor` | Color code for the timestamp of the remote user bubble. |  | #5B5C5E |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `remoteUserTypingTintColor` | `UIColor` | Color of the remote user typing bubbles animation. | | UIColor.white |
| `userBubbleBackgroundColor` | `UIColor` | Color code for the background of the visitor bubble. |  | #EDEDED |
| `userBubbleBorderColor` | `UIColor` | Color code for the outline color. |  | #EDEDED |
| `userBubbleLinkColor` | `UIColor` | Color code for links in the text of the visitor bubble. | | #0000ee |
| `userBubbleTextColor` | `UIColor` | Color code for the text of the visitor bubble. |  | UIColor.black |
| `userBubbleBorderWidth` | `Double` | Double number for the outline width. |  | 1 |
| `userBubbleTimestampColor` | `UIColor` | Color code for the timestamp of the visitor bubble. |  | #5B5C5E |
| `userBubbleSendStatusTextColor` | `UIColor` | Color code for the send status text of the visitor bubble. |  | #5B5C5E |
| `userBubbleErrorTextColor` | `UIColor` | Color code for the error view text of the visitor bubble. |  | #DE0A23 |

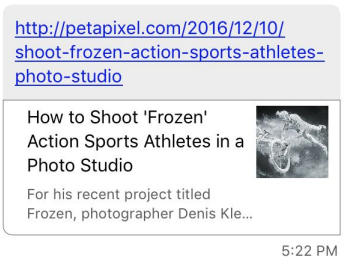
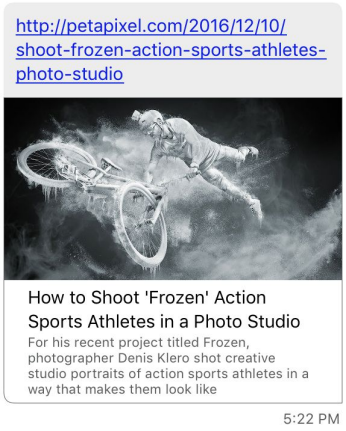
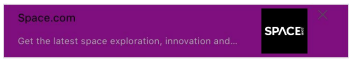
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `userBubbleErrorBorderColor` | `UIColor` | Color code for the error view border of the visitor bubble. |   | #DE0A23 |
| `bubbleEmailLinksRegex` | `String?` | Regular expression for email hyperlinks in users messages (consumer and agent). This attribute is optional - If not assigned, the default link detection will be enabled | | nil |
| `bubbleUrlLinksRegex` | `String?` | Regular expression for url hyperlinks in users messages (consumer and agent). This attribute is optional - If not assigned, the default link detection will be enabled | | nil |
| `bubblePhoneLinksRegex` | `String?` | Regular expression for phone hyperlinks in users messages (consumer and agent). This attribute is optional - If not assigned, the default link detection will be enabled | | nil |

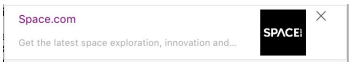
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| **Link Preview** | | | | |
|---|---|---|---|---|
| `enableLinkPreview` | `Bool` | Enable or disable link preview feature. If disabled, user will not see site's link preview or link preview. | | true |
| `linkPreviewBackgroundColor` | `UIColor` | Color code for the background of the link preview area inside cell. | | UIColor.white |
| `linkPreviewTitleTextColor` | `UIColor` | Color code for the title text inside link preview area inside cell. | | UIColor.black |
| `linkPreviewDescriptionTextColor` | `UIColor` | Color code for the description text inside link preview area inside cell. | | #5B5C5E |
| `linkPreviewSiteNameTextColor` | `UIColor` | Color code for the description site name link preview area inside cell. | | #E2E3E3 |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

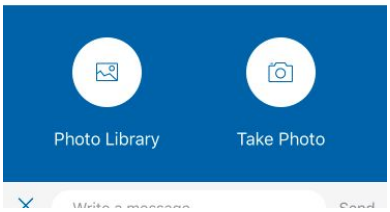
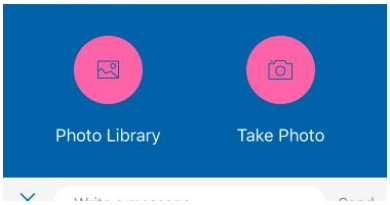
| | | | | |
|---|---|---|---|---|
| `linkPreviewBorderWidth` | `Double` | Double number for the outline width of link preview area inside cell. | | 1.0 |
| `linkPreviewStyle` | `LPUrlPreviewStyle` | Refers to the style in which the link preview cell will be displayed | Slim:<br>  <br>Large:<br>  | LPUrlPreviewStyle<br><br>.slim |
| `linkPreviewSiteNameTextColor` | `UIColor` | Color code for the description site name link preview area inside cell. | | |
| `urlRealTimePreviewBackgroundColor` | `UIColor` | The background color of the url real time preview |  | UIColor.white |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

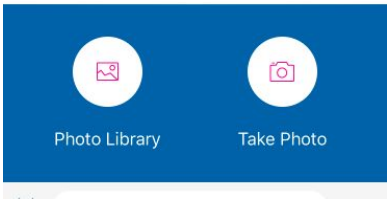
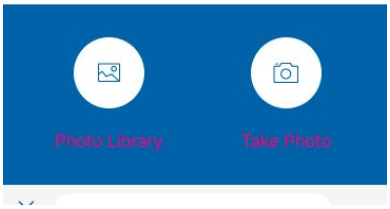
| | | | | |
|---|---|---|---|---|
| urlRealTimePreviewBorderColor | UIColor | The border color of the url real time preview | | |
| urlRealTimePreviewBorderWidth | CGFloat | The border width of the url real time preview | | |
| urlRealTimePreviewTitleTextColor | UIColor | The title text color of the url real time preview |  | |
| urlRealTimePreviewDescriptionTextColor | UIColor | The description text color of the url real time preview | | |
| useNonOGTagsForLinkPreview | Bool | urlPreview will also use non og tags to parse urls instead of using only og tags if useNonOGTagsForLinkPreview is true | | true |

| **Photo Sharing** | | | | |
|---|---|---|---|---|
| enablePhotoSharing | Bool | True<br><br>Enables Photo Sharing feature<br><br>False<br><br>Disables Photo Sharing | | false |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `maxNumberOfSavedFilesOnDisk` | `Int` | This number represents how many files will be saved on the disk. Exceeding files are deleted when the app closes. | | 20 |
| `fileCellLoaderFillColor` | `UIColor` | Radial loader fill color | | UIColor(white: 0.0, alpha: 0.5) |
| `fileCellLoaderRingProgressColor` | `UIColor` | Radial loader progress color | | UIColor.white |
| `fileCellLoaderRingBackgroundColor` | `UIColor` | Radial loader progress background color | | UIColor.lightGray |
| `photosharingMenuBackgroundColor` | `UIColor` | Photo Sharing menu background color |  | #0362AC |
| `photosharingMenuButtonsBackgroundColor` | `UIColor` | Photo Sharing menu buttons background color |  | UIColor.white |

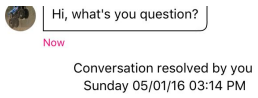
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

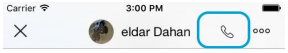
| | | | | |
|---|---|---|---|---|
| `photosharingMenuButtonsTintColor` | `UIColor` | Photo Sharing menu buttons tint color |  | #0362AC |
| `photosharingMenuButtonsTextColor` | `UIColor` | Photo Sharing menu buttons text color |  | UIColor.white |
| Note: Camera button, in the input text view, disabled/enabled color will change according to `sendButtonDisabledTextColor` and `sendButtonEnabledTextColor` configurations. | | | | |

| **Send Button** | | | | |
|---|---|---|---|---|
| `sendButtonDisabledColor` | `UIColor` | Color code for Send and Camera (of Photo Sharing) buttons in disabled mode. | | #AAAAAA |
| `sendButtonEnabledColor` | `UIColor` | Color code for Send and Camera (of Photo Sharing) buttons in disabled mode. | | #0362AC |
| `isSendMessageButtonInTextMode` | `Bool` | Two options for send message button mode:<br><br>Send message button in "text mode" | | Default mode is text |

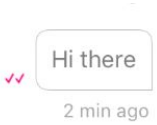
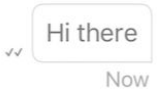
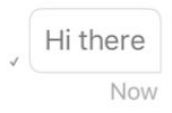
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | - will be taken from localized resources<br><br>Send message button in "icon mode" | | |

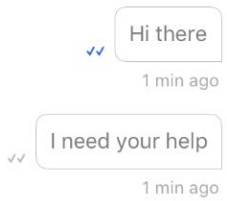
| **System Messages** | | | | |
|---|---|---|---|---|
| `systemBubbleTextColor` | `UIColor` | Color code for the text of the system messages. |  | UIColor.black |

| **Window Mode** | | | | |
|---|---|---|---|---|
| `customButtonImage` | `UIImage` | In window mode only:<br><br>Custom button icon image. This will be displayed on the navigation bar.<br><br>When the button is pressed, a dedicated callback will be invoked. (See LPMessagingSDKCustomButtonTapped for more information). |  | nil |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

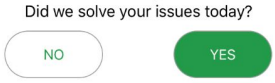
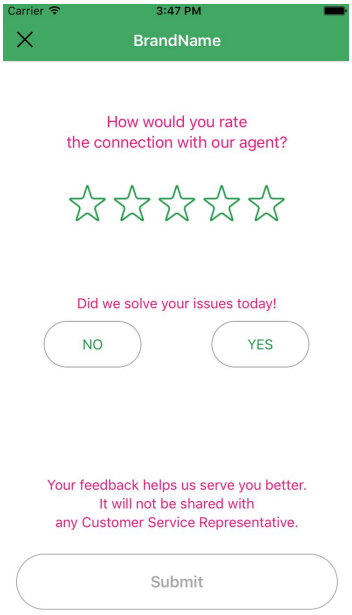
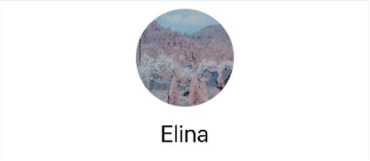
| **Delivery Notifications** | | | | |
|---|---|---|---|---|
| `checkmarkVisibility` | `CheckmarksState`<br><br>`(Integer Enum)` | Checkmark visibility of the following options (type `CheckmarksState`):<br><br>`SentOnly` - Show checkmarks for only Sent messages.<br><br>`SentAndAccepted` - Show checkmarks for only Sent and Accepted messages.<br><br>`All` - Show checkmarks for Sent, Accepted and Read messages. | ### | Checkmarks State<br><br>.All |
| `checkmarkReadColor` | `UIColor` | Color of checkmark indication signs of Read messages. |  | #004DC9 |
| `checkmarkDistributedColor` | `UIColor` | Color of checkmark indication signs of Distributed messages. |  | #5B5C5E |
| `checkmarkSentColor` | `UIColor` | Color of checkmark indication signs of Sent messages. |  | #5B5C5E |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

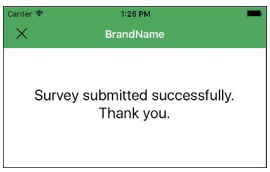
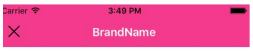
| | | | | |
|---|---|---|---|---|
| `isReadReceiptTextMode` | `Bool` | Two options for read indication:<br>• Read Receipt with Text Mode<br>• Read Receipt with Icon Mode<br>If the parameter set as true the mode will be Text.<br>If the parameter set as false the mode will be Icon.<br>Default value is true. |  | true |

| **Surveys Buttons (CSAT and FCR)** | | | | |
|---|---|---|---|---|
| `csatSubmitButtonCornerRadius` | `Double` | Corner radius of the Submit button. | | 30 |
| `csatSubmitButtonBackgroundColor` | `UIColor` | Background color code of the Submit button. | | #229A49 |
| `csatSubmitButtonTextColor` | `UIColor` | Text color code of the Submit button. | | UIColor.white |

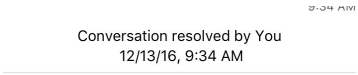
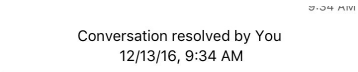
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| csatRatingButtonSelectedColor | UIColor | Background Color code of the rating buttons. | | #229A49 |
| csatResolutionButtonSelectedColor | UIColor | Color code for the FCR survey buttons (YES/NO) when selected. |  | #229A49 |
| csatAllTitlesTextColor | UIColor | Title text color for all labels. |  | UIColor.black |
| csatResolutionHidden | Bool | Hides the FCR survey (YES/NO) question.c | | false |
| csatAgentViewHidden | Bool | Hides the view of agent avatar and name. |  | true |

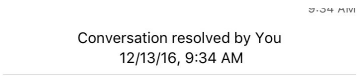
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

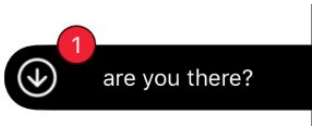
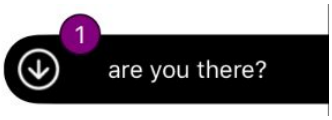
| | | | | |
|---|---|---|---|---|
| `csatThankYouScreenHidden` | `Bool` | Hides the Thank You screen after tapping Submit button. |  | false |
| `csatNavigationBackgroundColor` | `UIColor` | Background color of the navigation of the survey. |  | #229A49 |
| `csatNavigationTitleColor` | `UIColor` | Text color of the title in the survey navigation. | | UIColor.white |
| `csatSkipButtonColor` | `UIColor` | Skip survey button color. | | UIColor.black |
| `csatUIStatusBarStyleLightContent` | `Bool` | Allow the UI status bar to take the color of the survey navigation bar color. | | true |
| `csatShowSurveyView` | `Bool` | Hides the whole survey view and disables it. | | true |
| `csatSurveyExpirationInMinutes` | `UInt` | Expiration of CSAT in minutes from the moment the conversation was ended. If Survey exceeded the expiration, it will not be presented to the user. Default 24H | | 1440 |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

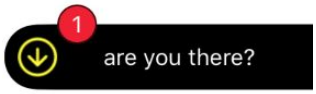
| **Conversations** | | | | |
|---|---|---|---|---|
| `maxPreviousConversationToPresent` | `UInt` | Number of conversations to show in advance. | | 2 |
| `deleteClosedConversationOlderThanMonths` | `UInt` | Upon SDK initialization, all closed conversations with an end date older than X months will be deleted from the database. Setting 0 will delete all closed conversations. | | 13 |
| `sendingMessageTimeoutInMinutes` | `UInt` | Maximum number of minutes to send the message | | 60 |
| `conversationSeparatorTextColor` | `UIColor` | Conversation separator text and line color |  | UIColor.black |
| `enableConversationSeparatorTextMessage` | `Bool` | Toggle conversation separator text message when conversation resolved from agent or consumer |  | true |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `enableConversationSeparatorLine` | `Bool` | Toggle conversation separator line when conversation resolved from agent or consumer |  | true |
| `enableVibrationOnMessageFromRemoteUser` | `Bool` | Toggle vibration sound when a new message from a remote user received | | false |

| **Unread Messages** | | | | |
|---|---|---|---|---|
| `scrollToBottomButtonBackgroundColor` | `UIColor` | Scroll to bottom button background color of the whole button |  | UIColor.black |
| `scrollToBottomButtonMessagePreviewTextColor` | `UIColor` | Scroll to bottom button text color of the last unread message preview | | UIColor.white |
| `scrollToBottomButtonBadgeBackgroundColor` | `UIColor` | Scroll to bottom button unread message badge background color |  | #E7242D |
| `scrollToBottomButtonBadgeTextColor` | `UIColor` | Scroll to bottom button unread message badge text color | | UIColor.white |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `scrollToBottomButtonArrowColor` | `UIColor` | Scroll to bottom button arrow tint color |  | UIColor.white |
| `unreadMessagesDividerBackgroundColor` | `UIColor` | Unread Messages divider background color | | #F5F5F5 |
| `unreadMessagesDividerTextColor` | `UIColor` | Unread Messages divider text color | | #004DC9 |
| `scrollToBottomButtonEnabled` | `Bool` | Toggle the mode of the Scroll to bottom button | | true |
| `scrollToBottomButtonMessagePreviewEnabled` | `Bool` | Toggle the mode of the Scroll to bottom unread message text preview | | true |
| `unreadMessagesDividerEnabled` | `Bool` | Toggle the mode of the Unread Messages divider. If disabled, scroll to bottom button will scroll to bottom although we can have new messages and don't show the badge at all nor "new message preview" | | true |

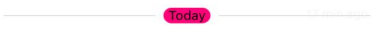
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| **Localization** | | | | |
|---|---|---|---|---|
| `country` | `String?` | Country code: When it is not nil, it will be combined with 'language' ("language_country" , for example: en_US) and used instead of device default locale when formatting date and time. If no value is provided, the SDK will use the country according to the device's locale. | | nil |
| `language` | LPLanguage | Language that will be used instead of default device language.<br><br>Its type is LPLanguage enum that contains all the languages that are supported by MessagingSDK.<br><br>It will affect the following areas:<br><br>1. Will be used when getting localized strings<br><br>2. Will be combined with 'country' ("language_country" , for example: en_US) and used instead of default device locale when | | DeviceLanguage |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

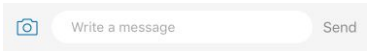
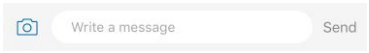
| | | | | |
|---|---|---|---|---|
| | | formatting time and date. If no value is provided, the SDK will use the device's language as default. | | |

| **Brand** | | | | |
|---|---|---|---|---|
| `brandName` | `String` | The brand name will be shown as a title on toolbar when there is no active conversation. | | "BrandName" |
| `conversationBackgroundColor` | `UIColor` | Color code for the entire view background. | | UIColor.white |

| **Date Separator** | | | | |
|---|---|---|---|---|
| `dateSeparatorTitleBackgroundColor` | `UIColor` | Color code for date separator title background color. |  | UIColor.white |
| `dateSeparatorTextColor` | `UIColor` | Color code for date separator text color. | | #46474A |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `dateSeparatorLineBackgroundColor` | `UIColor` | Color code for date separator line background color. | | UIColor.clear |
| `dateSeparatorBackgroundColor` | `UIColor` | Color code for date separator background color. | | #FFFFFF |

| **User input view** | | | | |
|---|---|---|---|---|
| `inputTextViewContainerBackgroundColor` | `UIColor` | User Input TextView container background color. |  | #F5F5F5 |
| `inputTextViewCornerRadius` | `Double` | User Input TextView corner radius. |  | 17.0 |

| **Agent Assignment** | | | | |
|---|---|---|---|---|
| `retrieveAssignedAgentFromLastClosedConversation` | `Bool` | When using “getAssignedAgent” method, this option lets you decide whether to get assigned agents from active conversations only, or also from the last closed conversation in case there is no active conversation. If not assigned | | true |

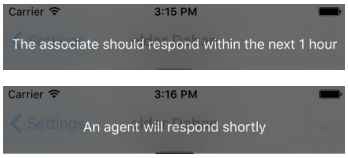
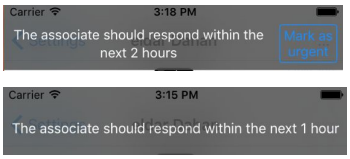
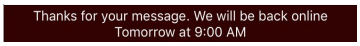
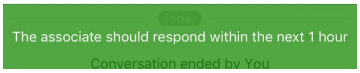
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | agent is available this method will return nil. | | |

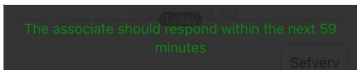
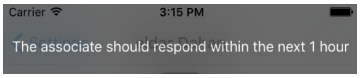
| **Duration of Local Notifications** | | | | |
|---|---|---|---|---|
| `notificationShowDurationInSeconds` | `Double` | Display duration of the local notification in seconds. Examples: TimeToRespond notification, local notification, etc.<br><br>Note: this parameter will be extended to 60sec when in VoiceOver mode. | | 3 (60 when in VoiceOver mode) |

| **Time To Response and Off hours** | | | | |
|---|---|---|---|---|
| `ttrShowShiftBanner` | `Bool` | Ability to enable/disable shift toaster (‘An agent will respond...’) | | true |
| `ttrFirstTimeDelay` | `Double` | TTR - Time To Respond. | | 10 |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | Number of seconds before the first TTR notification appears. | | |
| `ttrShouldShowTimestamp` | `Bool` | TTR - Time To Respond.<br><br>**Enable:**<br><br>Displays a time stamp in the TTR notification.<br><br>**Disable:**<br><br>Displays: “An agent will respond shortly”. |  | false |
| `ttrShowFrequencyInSeconds` | `UInt` | Controls the TTR frequency: Don’t show the TTR more than once in X seconds. | | 8 |
| `showUrgentButtonInTTRNotification` | `Bool` | TTR - Time To Respond<br><br>Enable presentation of Urgent button in the TTR notification. |  | false |
| `showOffHoursBanner` | `Bool` | Ability to disable/enable the off-hours toaster. |  | true |
| `ttrBannerBackgroundColor` | `UIColor` | Color of background for banner. |  | #52A742 |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| `ttrBannerTextColor` | `UIColor` | Text color of the banner. |   | #52A742 |
| `ttrBannerOpacityAlpha` | `Double` | Opacity level of the banner background (values: 0.0 - 1.0). |   | 0.8 |
| `offHoursTimeZoneName` | `String` | Off Hours time zone name string based on `[NSTimeZone knownTimeZoneNames].`<br><br>If sending empty string, the local timezone will be used (Server sends UTC time). | | "" |

| **Date and Time** | | | | |
|---|---|---|---|---|
| `lpDateFormat` | `String?` | Custom formatting for date string (day, year..), for example: 'd MMM'.<br><br>If not defined, one of the default styles will be used (see timestamps formatting). | | nil |
| `lpTimeFormat` | `String?` | Custom formatting for time string (hours, lpDateTimeFormat | | nil |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | minutes..), for example: 'hh:mm a'.<br><br>If not defined, one of the default styles will be used (see timestamps formatting). | | |
| `lpDateTimeFormat` | `String?` | Custom formatting for date and time string, for example: 'EEEE MM/dd/YY hh:mm a'.<br><br>If not defined, one of the default styles will be used (see timestamps formatting). | | nil |

| **Toast Notifications** | | | | |
|---|---|---|---|---|
| `toastNotificationsEnabled` | `Bool` | Enable toast notifications such as offline and TTR notifications<br><br>**False:**<br><br>Disable toast notifications. | | true |
| `csdsDomain` | `String` | CSDS Domain URL. For brands that need to control the URL that is the gateway for | "https://adminlogin.liveperson.net/csdr/account/%@/service/baseURI.json?version=1.0" | |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | LivePerson services, use this key to set a URL of your choice. | | |

| **User Avatar** | | | | |
|---|---|---|---|---|
| `remoteUserAvatarBackgroundColor` | `UIColor` | Background color of the remote user's avatar. | | #004DC9 |
| `remoteUserAvatarIconColor` | `UIColor` | Icon color of default remoteUser avatar. | #0362AC | #FFFFFF |
| `brandAvatarImage` | `UIImage?` | Set avatar image for brand. This is an optional UIImage that if is set to nil a default avatar will be presented.<br><br>Image ratio should be 1:1 (square) and at least 50x50 pixels. | | nil |
| `csatAgentAvatarBackgroundColor` | `UIColor` | Background color of agent's default avatar in CSAT. | | #004DC9 |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| csatAgentAvatarIconColor | UIColor | Icon color of agent's default avatar in CSAT. | | #FFFFFF |

| **Data Masking** | | | | |
|---|---|---|---|---|
| enableClientOnlyMasking | Bool | Determines whether to enable using regular expression to control which part of the text to mask, all masked data will appear as asterisks, will be saved to local db masked and will be sent to the server unmasked. | | false |
| enableRealTimeMasking | Bool | Determines whether to enable using regular expression to control which part of the text to mask, all masked data will appear as asterisks, will be saved to local db masked and sent to the server masked. | | false |
| clientOnlyMaskingRegex | String | Regular expression string to control which part of the text to mask, all masked data will | | "" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | | | |
|---|---|---|---|---|
| | | appear as asterisks, will be saved to local db masked and will be sent to the server unmasked. Default is empty, meaning no regex.<br><br>The regular expression patterns and behavior are based on **Perl's** regular expressions. See<br><br>Apple Reference. | | |
| `realTimeMaskingRegex` | `String` | Regular expression string to control which part of the text to mask. All masked data will appear as asterisks, will be saved to local db masked, and will be sent to the server unmasked. Default is empty, meaning no regex.<br><br>The regular expression patterns and behavior are based on **Perl's** regular expressions. See<br><br>Apple Reference. | | `""` |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| Navigation | | | | |
|---|---|---|---|---|
| conversationNavigationBackgroundColor | UIColor | Background color of navigation bar in conversation screen. | | #0362AC |
| conversationNavigationTitleColor | UIColor | Navigation title color in conversation screen. | | #FFFFFF |
| conversationStatusBarStyle | UIStatusBarStyle | Status bar style in conversation screen. | | .LightContent |

## Deprecated Attributes

Listed below are attributes that have been deprecated. These keys should no longer be used as LPConfig files, but rather as Localised strings keys. The names of these keys in the Localisation files remain the same as in the LPConfig files.

In order to configure different string for this key, see String localization in SDK.

*Note: By default, these keys are configured with Localization keys values, and are sensitive to any language changes. These keys must be reconfigured each time a language is changed.*

| Attribute name | Type | Description |
|---|---|---|

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | | |
|---|---|---|
| `customButtonDescription` | `String` | Accessibility voiceover string for the custom button. |
| `readReceiptTextDistributed` | `String` | Text for distributed indication. |
| `csatResolutionFeedbackText` | `String` | Text for the feedback label. |
| `csatResolutionQuestionText` | `String` | Text for the resolution confirmation question. |
| `readReceiptTextSent` | `String` | Text for sent indication. |
| `readReceiptTextRead` | `String` | Text for read indication. |
| `readReceiptTextSending` | `String` | Text for sending indication. |
| `sendButtonDisabledTextColor` | `UIColor` | Send button color in disabled mode in the conversation screen |
| `sendButtonEnabledTextColor` | `UIColor` | Send button color in enabled mode in the conversation screen |
| `editTextUnderlineColor` | `UIColor` | User text underline color |

## String localization in SDK

The SDK contains language folders for each language supported. For a list of supported languages, see LiveEngage System Requirements and Language Support. Each folder contains LPLocalizable.strings file, where all strings are located for a specific language.

The example below contains four language folders:

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- en.lproj: strings in English. Used in case the host app/device language is not supported by us.
- fr.lproj: strings in French.
- pt.lproj: strings in Portuguese.
- pt-PT.lproj: strings in Portuguese (Portugal).

The SDK allows you to override the string localization of any supported language in LiveEngage.

To apply a custom localization files with your own strings, create LPLocalizable.strings files for relevant languages inside your app, which will include the keys you would like to override.

Example: Overriding the SDK string of 'send' in English:

1. Create in your app a new localization Base file called *LPLocalizable.strings* which will include your supported language: 'New file…' -> 'Strings file' -> 'Create'
2. Add new key: "send" = "<ANY NEW VALUE>";
3. Mark this *LPLocalizable.strings* file as localized: Tap on *LPLocalizable.strings file -> Open the file inspector -> Go to* 'Localization' section -> Tap 'Localize…'
4. Adding a new supported language
   - Open project settings
   - Info tab
   - Under Localizations, press '+' sign
   - Choose the new language you would like to support.

   Attaching a Strings file to existing supported languages:
   - Choose the desired strings file
   - Open the file inspector
   - Under localizations, choose the relevant languages.

Your 'send' string implementation will override the localization in English in the SDK

In order to print all localized keys with a default English value in the SDK, call the following:

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

```
LPMessagingSDK.instance.printAllLocalizedKeys()
```

In order to get or print all localized supported languages in the SDK, call the following:

```
LPMessagingSDK.instance.printSupportedLanguages()
```

In order to get all supported languages by key-value (key is locale key and value is language description), call the following:

```
LPMessagingSDK.instance.getAllSupportedLanguages()
```

*Note: The SDK uses native iOS control for the Copy/Paste menu. The language of the menu control is defined by the current device language if the host app is localized for this language. If the host app does not have this language, the last menu control language will be used.*

## Localization Keys

Listed below are all localized keys with the English localization string.

| **Attribute name** | **String Localization** |
|---|---|
| `customButtonAccessibilityDescription` | "Call" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| readReceiptTextDistributed | “Distributed” |
| conversationEndedByAgent | "Conversation resolved by %@ \n %@" |
| conversationEndedByAgentWithoutName | "Conversation resolved by Agent \n %@" |
| conversationEndedByYou | "Conversation resolved by You \n %@" |
| cancel | "Cancel" |
| hiMessage | "How can I help you today?" |
| sentMessageErrorTitle | "Message was not sent" |
| retry | "Retry" |
| sentMailErrorTitle | "Mail not sent" |
| sentMailErrorMessage | "Email could not be sent. Please try again later." |
| callTo | "Call %@" |
| sendMessage | "Send Message" |
| facetimeAudio | "FaceTime Audio" |
| sendEmailTitle | "Send Email" |
| sendEmailMessage | "Would you like to send an email to %@?" |
| notNow | "Not now" |
| yes | "Yes" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| `feedbackYesButtonTitle` | "Yes" |
| `feedbackNoButtonTitle` | "No" |
| `cantSendMail` | "Can't send email" |
| `noEmailOnDeviceError` | "No email is defined on this device" |
| `dismissUrgent` | "Dismiss urgency" |
| `dismissUrgentConfirmation` | "Are you sure you want to mark this conversation as not urgent?" |
| `dismiss` | "Dismiss" |
| `markAsUrgent` | "Mark as urgent" |
| `markAsUrgentConfirmation` | "Are you sure you want to mark this conversation as urgent?" |
| `urgent` | "Urgent" |
| `noInternetConnection` | "No Internet Connection" |
| `ok` | "OK" |
| `markAsEndedConfirmation` | "Are you sure this topic is resolved?" |
| `endTheConversation` | "Resolve the conversation" |
| `ttrResponseMessage` | "An agent will respond within the next %@" |
| `rateConnectionWithAgent` | "How would you rate your\nconnection with our agent?" |
| `rateConnectionWithName` | "How would you rate your\nconnection with %@?" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| `veryDissatisfied` | "Very Dissatisfied" |
| `dissatisfied` | Dissatisfied |
| `neither` | "Neither" |
| `satisfied` | "Satisfied" |
| `verySatisfied` | "Very Satisfied" |
| `writeMessage` | "Write a message" |
| `olderMessages` | "Older messages" |
| `numberOfDays` | "%d days" |
| `day` | "1 day" |
| `numberOfHours` | "%d hours" |
| `oneHour` | "1 hour" |
| `oneMinute` | "1 minute" |
| `numberOfMinutes` | "%d minutes" |
| `minutesAgo` | "%d min ago" |
| `send` | "Send" |
| `submit` | "Submit" |
| `today` | "Today" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| noSocketConnectivity | "No connectivity to server\nPlease try again" |
| conversationAlreadyEnded | "This conversation has already been resolved." |
| loadingHistoryMessages | "Loading..." |
| and | "and" |
| anAgentWillRespondShortly | "An agent will respond shortly" |
| nowTimestamp | "Now" |
| surveySubmittedSuccessfullyThankYou | "Survey submitted successfully.\nThank you!" |
| ttrOffHoursResponseMessageToday | "Thanks for your message. We will be back online Today at %@" |
| ttrOffHoursResponseMessageTomorrow | "Thanks for your message. We will be back online Tomorrow at %@" |
| ttrOffHoursResponseMessageDate | "Thanks for your message. We will be back online %@" |
| maskedMessageResendError | "Message failed to send. Please re-enter message and send again." |
| systemMessageRealTimeMasked | "Your personal data has been masked to protect your security and cannot be read by the agent." |
| systemMessageClientOnlyMasked | "Your personal data has been masked to protect your security. Only the agent can read it." |
| offlineNotification | "Offline. Check your internet connection." |
| resolve | "Resolve" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| unreadMessages | "%@ UNREAD MESSAGES" |
| invalidHyperLinkError | "Link is not valid" |
| accessibilityAgentDefaultAvatar | “Agent” |
| accessibilityAgentIsTyping | "%@ is typing" |
| accessibilityAgentWithoutNameIsTyping | "Agent is typing" |
| accessibilityMessageFromAgentWithName | "Agent %@: %@" |
| accessibilityMessageFromAgentWithoutName | "Agent: %@" |
| accessibilityMessageFromUser | "You: %@" |
| accessibilityCameraButton | “Camera” |
| accessibilityCloseScreenButton | "Close" |
| accessibilityNavigationMenuButton | "Menu" |
| accessibilityCloseCameraButton | "Close camera" |
| accessibilityMessageError | "Error" |
| accessibilityMessageErrorHint | "Tap to resend message" |
| accessibilityFileUploadAnnouncement | "Uploading photo" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| `accessibilityFileDownloadAnnouncement` | "Downloading photo" |
| `accessibilityPhotoMessageFromAgentWithName` | "Agent %@: photo" |
| `accessibilityPhotoMessageFromAgentWithoutName` | "Agent: photo" |
| `accessibilityPhotoMessageFromUser` | "You: photo" |
| `accessibilityPhoto` | "Photo" |
| `accessibilityFileNoFileState` | "Double tap to download" |
| `accessibilityFileUploadingState` | "Uploading" |
| `accessibilityFileDownloadingState` | "Downloading" |
| `accessibilityFileSyncedState` | "Double tap to open" |
| `accessibilityFileDownloadedErrorState` | "Failed to download" |
| `accessibilityFileNotAvailableState` | "Not available" |
| `accessibilityScrollIndicator` | "Scroll indicator" |
| `accessibilityRealTimeLinkPreviewView` | "Link preview" |
| `accessibilityRealTimeLinkPreviewClosePreviewButton` | "Close link preview" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| `accessibilityRealTimeLinkPreviewCloseButtonHint` | "Press to close link preview" |
| `accessibilityRealTimeLinkPreviewTitle` | "Link preview title" |
| `accessibilityRealTimeLinkPreviewDescription` | "Link preview description" |
| `accessibilityRealTimeLinkPreviewImage` | "Link preview image" |
| `clearHistoryMenuTitle` | “Clear history” |
| `clearHistoryConfirmation` | "All of your existing conversation history will be lost. Are you sure?" |
| `clearHistoryFailureMessage` | "Please resolve the conversation first." |
| `clearHistoryConfirmButton` | "Clear" |
| `sharePhotoFromPhotoLibrary` | "Photo Library" |
| `sharePhotoFromCamera` | "Take Photo" |
| `tapToAddACaption` | "Tap to add a caption" |
| `errorSendingThePhoto` | "Error sending the photo" |
| `sorryWeveEncounteredAnErrorWhileSendingThePhoto` | "Sorry, we've encountered an error while sending the photo. Please try again." |
| `readReceiptTextSending` | "Sending" |

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

| | |
|---|---|
| `readReceiptTextSent` | "Sent" |
| `readReceiptTextDistributed` | "Distributed" |
| `readReceiptTextRead` | "Read" |

## Timestamps Formatting

### Time/Date Styles

The iOS platform provides four different default types of date and time styles:

SHORT is completely numeric.
MEDIUM is longer and contains the first 3 letters of the month.
LONG is longer and contains the full month name.
FULL specifies the complete time and date.

iOS examples:

SHORT 12/13/52 or 3:30 PM
MEDIUM Jan 12, 1952
LONG January 12, 1952 or 3:30:32 PM
FULL Tuesday, April 12, 1952 AD or 3:30:42 PM PST.

The LPMessagingSDK uses default styles. Each feature has its own style. The style is flexible and adapts the 'locale' configuration of the device.

Example: US locale SHORT date is displayed as "9/25/16", whereas Japanese locale SHORT date is displayed as "2016/9/25".

A specific 'locale' which is different from the device locale can be set through the language and country configurations.

Example: country: String?, language: String?

Time/Date formats:

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

It is also possible to configure a specific format for time and date instead of style.

If the host app has configured its own formatting, this formatting will be used instead of style (and therefore will not be affected by 'locale').

To configure the formatting by the host app, three configurable formatting resources have been added:

1. For date only (separator):
   lpDateFormat: String?
2. For time only (bubble's timestamp & off hours time in case of today/tomorrow):

   lpTimeFormat: String?

3. For date & time together (resolve message & off hours time in case of other date):

   lpDateTimeFormat: String?

## Off Hours

*Date & Time*

- The Today and tomorrow off hours message uses the default SHORT time without date according to the locale (default or custom).

If the device is set to 12 hour format :

> "Thanks for your message. We will be back online today/tomorrow at *3:30 PM*"

If the device is set to 24 hour format :

> "Thanks for your message. We will be back online today/tomorrow at *15:30*"

To use a different time format, you can use `lpTimeFormat: String?` with any time format.

Example: "hh:mm a", "HH:mm", etc.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

- The Date off hours message (not today/tomorrow) uses default LONG date and SHORT time according to the locale (default or custom).

If the device is set to 12 hour format :

> “Thanks for your message. We will be back online *January 12, 2017* at *3:30 PM*”

If the device is set to 24 hour format :

> “Thanks for your message. We will be back online *January 12, 2017* at *15:30*”

To use a different date/hour format, you can use `lpDateFormat: String?` with any date & time format.

Example: “MMM d, yyyy hh:mm a”, “EEEE dd/mm/yy HH:mm” etc.

*Timezone*

Off hours can appear in different time zones with this resource ID :

```
offHoursTimeZoneName: String = ""
```

A list of timezone IDs can be found here.

Example: ”US/Pacific”, “Europe/Berlin”.

## Bubble timestamp

The Bubble timestamp contains the time only in the SHORT time format, according to the locale (default or custom) and to the device setting.

If the device is set to 12 hours format : “3:30 PM”

If the device is set to 24 hours format : ”15:30”

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

If you wish to configure this time format, override the resource ID `lpTimeFormat: String?` with any time format.

Example: “hh:mm a”, “HH:mm” etc.

*Note: This applies to all bubble timestamps.*

## Separator timestamp

The Separator timestamp contains the date only in the SHORT date format, according to the locale (default or custom) and to device setting.

“9/25/16” for US locale / “2016/9/25” for JP locale

If you wish to configure this time format, override the resource ID `lpDateFormat: String?` with any date format.

Example: “MMM d, yyyy”, “EEEE dd/mm/yy” etc.

## Resolve message

The Resolve message uses the default SHORT date and SHORT time according to the locale (default or custom) and to the device setting.

If the device is set to 12 hour format (US locale):

> “Conversation resolved by [agent name] \n 9/25/16, 3:30 PM”

If the device is set to 24 hour format (US locale):

> “Conversation resolved by [agent name] \n 9/25/16, 15:30”

To use a different date/hour format, you can use `lpDateTimeFormat: String?` with any date & time format.

Example: “MMM d, yyyy hh:mm a”, “EEEE dd/mm/yy HH:mm” etc.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

*Limitation: If the formatting is changed after the Resolve messages already appears, this change will take no effect. This will be fixed in future versions.*

# OS Certificate Creation

In order for the push notification to work, you will need two .pem files:

1. A certificate file stored using a pem format.
2. A Key file stored using a pem format without a password.

## Creating a Certificate Signing Request file

With this file, we will create both a .p12 file and a .crt file.

To manually generate a Certificate, you need a Certificate Signing Request (CSR) file from your Mac. To create a CSR file, follow the instructions below to create one using Keychain Access.

1. In the Applications folder on your Mac, open the Utilities folder, and launch Keychain Access.

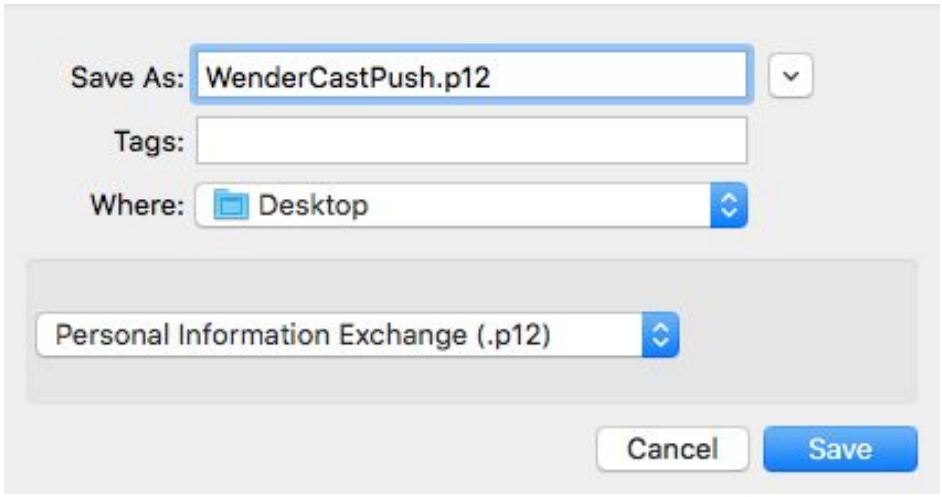
Within the Keychain Access dropdown menu, select **Keychain Access** > **Certificate Assistant** > **Request a Certificate from a Certificate Authority.**

2. In the Certificate Information window, enter the following information:
    - In the User Email Address field, enter your email address.
    - In the Common Name field, create a name for your private key, for example,John Doe Dev Key.
    - The CA Email Address field should be left empty.
    - In the Request is group, select the Saved to disk option.
3. In Keychain Access, click **Continue** to complete the CSR generating process.
4. Download and run the certificate. The certificate is now added to your Keychain, paired with a private key:

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

## Creating a key .p12 file

1. Right-click on your new push certificate and choose **Export**.

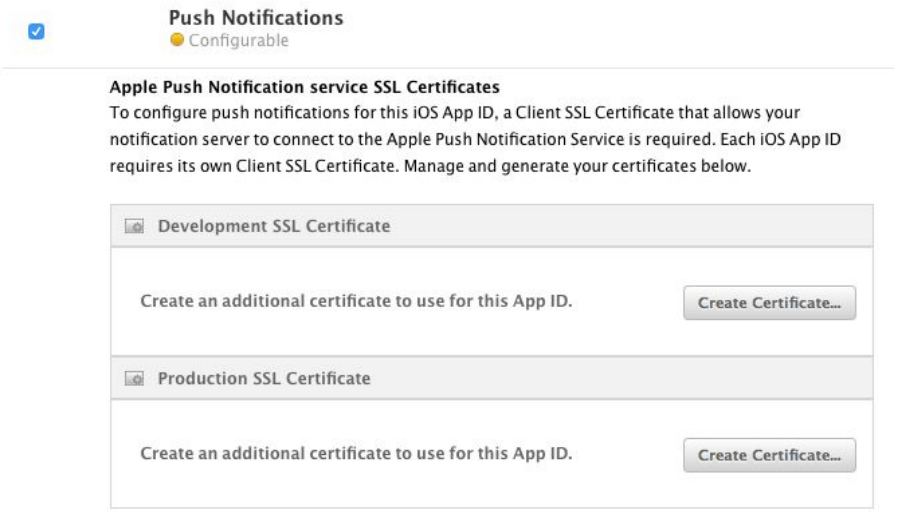



2. Save the certificate as pushNotification.p12 as a .p12 file.
3. You will be prompted to enter a password for the p12. You can either leave this blank or enter a password of your choosing.

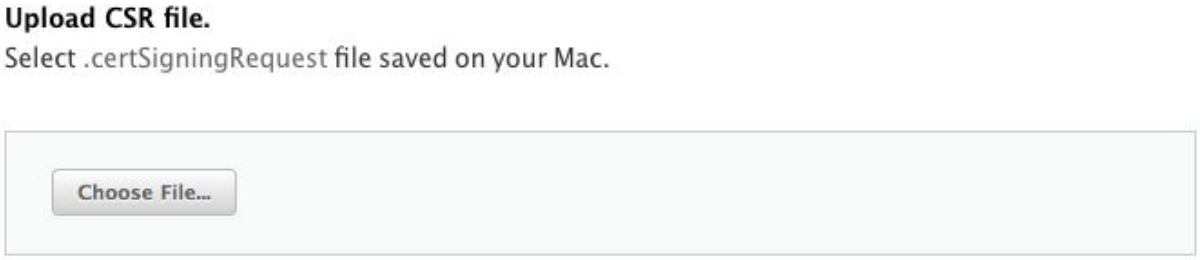
## Creating a push notification certificate. (der format .crt file)

1. In the iOS member area, go to your app area.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.




2. Under Push Notification service SSL Certificates, select **Create Certificate** (Developer or Production).




3. Choose the .csr file that you created in the previous stage.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

4. Download the file.This file will be used to create the .pem format certificate.

## Creating both key.pem file and cert.pem file

*Note: This is used when configuring LiveEngage Push Notification.*

1. Open the terminal and locate the folder in which you would like to save the file.

2. Create cert pem:

   **openssl x509 -in aps_development.cer -inform der -out cert.pem**

3. Convert the private keys .p12 file into a .pem file:

   **openssl pkcs12 -nocerts -out keyWithPassword.pem -in key.p12**

   You will be prompted to enter a passphrase for this file. Enter any password and remember it for the next step.

4. RSA .pem key (no password)

   **openssl rsa -in hostkey.pem -out hostkey.pem**

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

You will be prompted to enter a passphrase. Enter the same passphrase you used in the previous step.

# CSAT Behavior

## Overview

This document describes the CSAT behaviour and configurations in the Messaging SDK.
You can find all the related configurations in the resources ID table, under Survey Screen.

## Show CSAT flow

Show if:

1. CSAT configured to appear according to LPConfig.defaultConfiguration.csatShowSurveyView
2. Conversation has an assigned agent.
3. Conversation's CSAT wasn't previously submitted.

## Dismiss CSAT

The CSAT view is dismissed in one of four cases:

1. User pressed the submit button (answers are sent to the survey).
2. User choose to skip the CSAT (skipped button is pressed).
3. The CSAT is automatically dismissed if it was filled in any other device.
4. If agent resumed the conversation while csat is visible - it will automatically dismissed.

## CSAT UI content

CSAT screen includes several content containers:

Agent View (avatar and agent name)

1. Could be hidden or not according to LPConfig.defaultConfiguration.csatAgentViewHidden
2. Contains agent avatar:
    a. If conversation has assigned agent and its image was downloaded previously using profileUrl, this image will be presented in the view.
    b. If no image available, default avatar is presented. It's background and tint color is according to agent bubble with LPConfig.defaultConfiguration.csatAgentAvatarBackgroundColor and LPConfig.defaultConfiguration.csatAgentAvatarIconColor
3. Contains agent name:
    a. By default it's an empty label.
    b. If conversation has assigned agent, the agent's nickName will be used.

Rating Question View (stars)

1. Always visible - can't configure its visibility.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

2. Stars color is defined by LPConfig.defaultConfiguration.csatRatingButtonSelectedColor

3. Rating question includes 'Agent' by default in the text. If conversation has assigned agent and the agent's nickName is not empty, this nickName will be used instead.

Resolution Confirmation View (yes/no)

1. Could be hidden or not according to LPConfig.defaultConfiguration.csatResolutionHidden

2. If agentView is shown ("csatAgentViewHidden"), this view will be always hidden (even if "csatResolutionHidden" is set to true)

3. All titles colors defined with LPConfig.defaultConfiguration.csatAllTitlesTextColor

# Photo Sharing (Beta)

## Overview

This section describes the photo sharing behavior and configurations in the Messaging SDK.

You can find all the related configurations in the resources ID table, under Photo Sharing.

*Notes:*

- *This feature is available only for the In-App Messaging SDK.*
- *This features enables photo sharing only (not video/files).*
- *Photo-sharing is one-way only: Photos can be sent from consumer to agent, but not vice versa.*
- *Device storage includes up to 20 images - this is configurable.*
- *Supported formats: .png, .jpg, .gif (non-animated).*
- *Photo size reduction: Thumbnail - 30 KB, Preview -3 MB.*

## Enable Photo Sharing

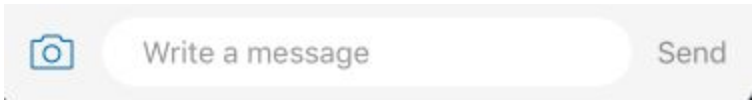
To enable/disable photo sharing you can change the boolean value LPConfig.defaultConfiguration.enablePhotoSharing

By default this value is set to false.

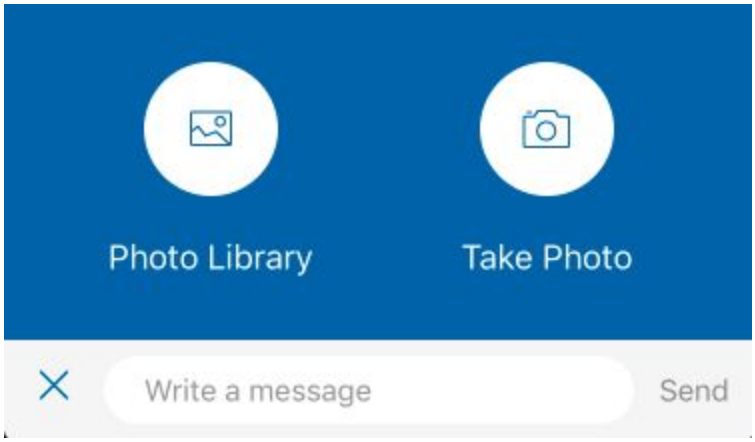
## Upload Photo

To upload a photo, click the "attach" button next to "enter message" edit text.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.




A menu will open with 2 options: Photo Library and Camera.




Changing the background color of attachment menu is available with configuration :
LPConfig.defaultConfiguration.photosharingMenuBackgroundColor

Changing the text of buttons:
LPConfig.defaultConfiguration.photosharingMenuButtonsTextColor
Changing the menu button's background color:
LPConfig.defaultConfiguration.photosharingMenuButtonsBackgroundColor
Changing the menu button's tint color:

LPConfig.defaultConfiguration.photosharingMenuButtonsTintColor

## Configuring Push Notifications

Follow the instructions below to set up your app key to enable push notifications.

*Note: Before you begin the setup, you must ensure your LiveEngage account is configured and connected to the SDK.*

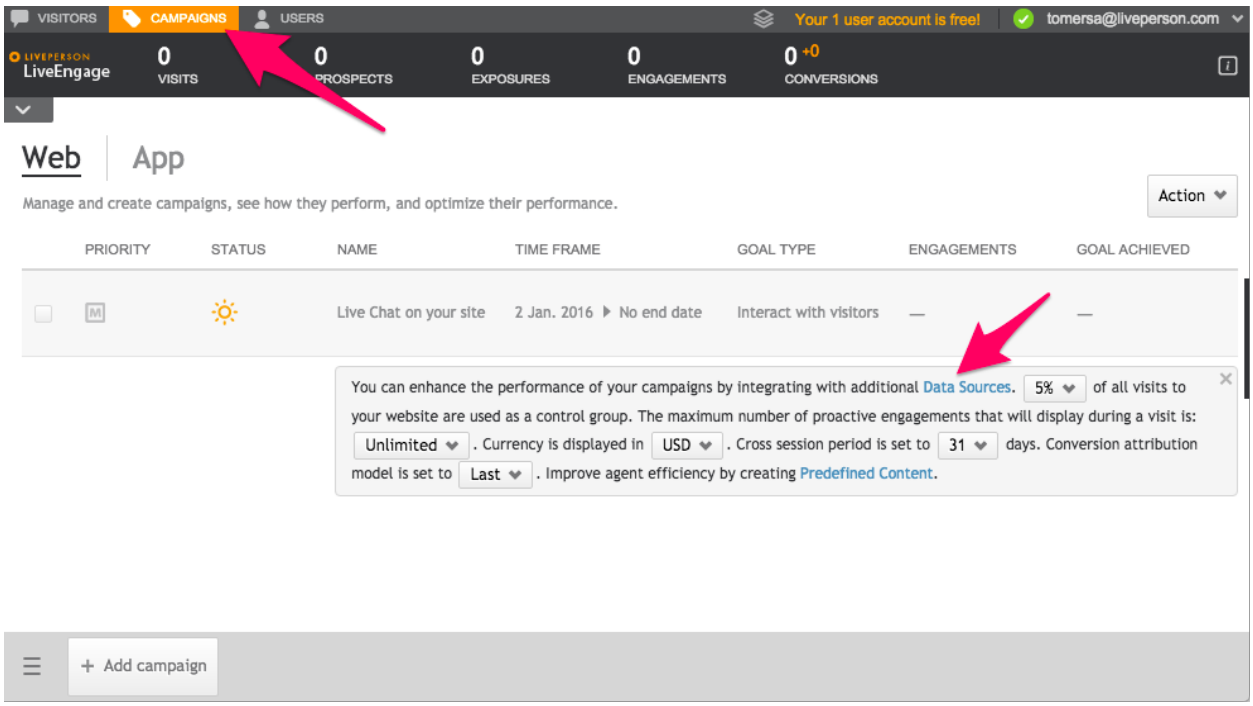
1. Enter your LiveEngage account through this Login URL.

   You will need the following info from your LivePerson account team:

- LiveEngage account number
- User ID (must be an administrator user)
- Password

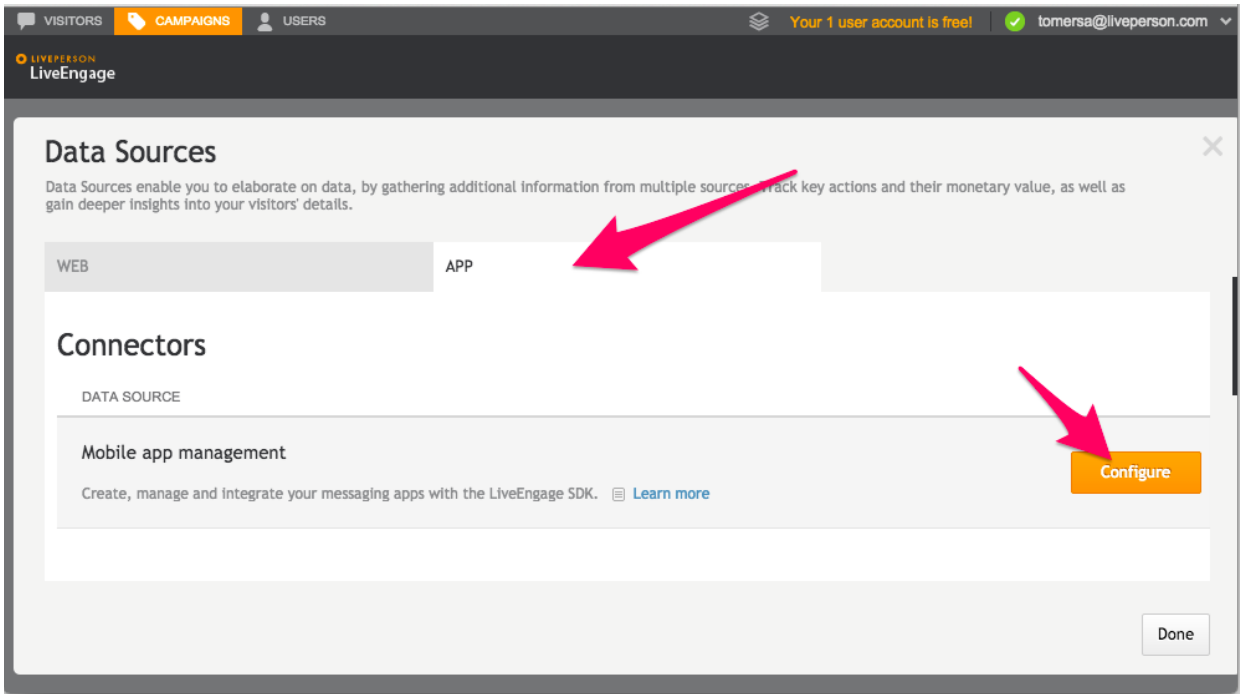
**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

2. Within LiveEngage, navigate to **Campaigns.**

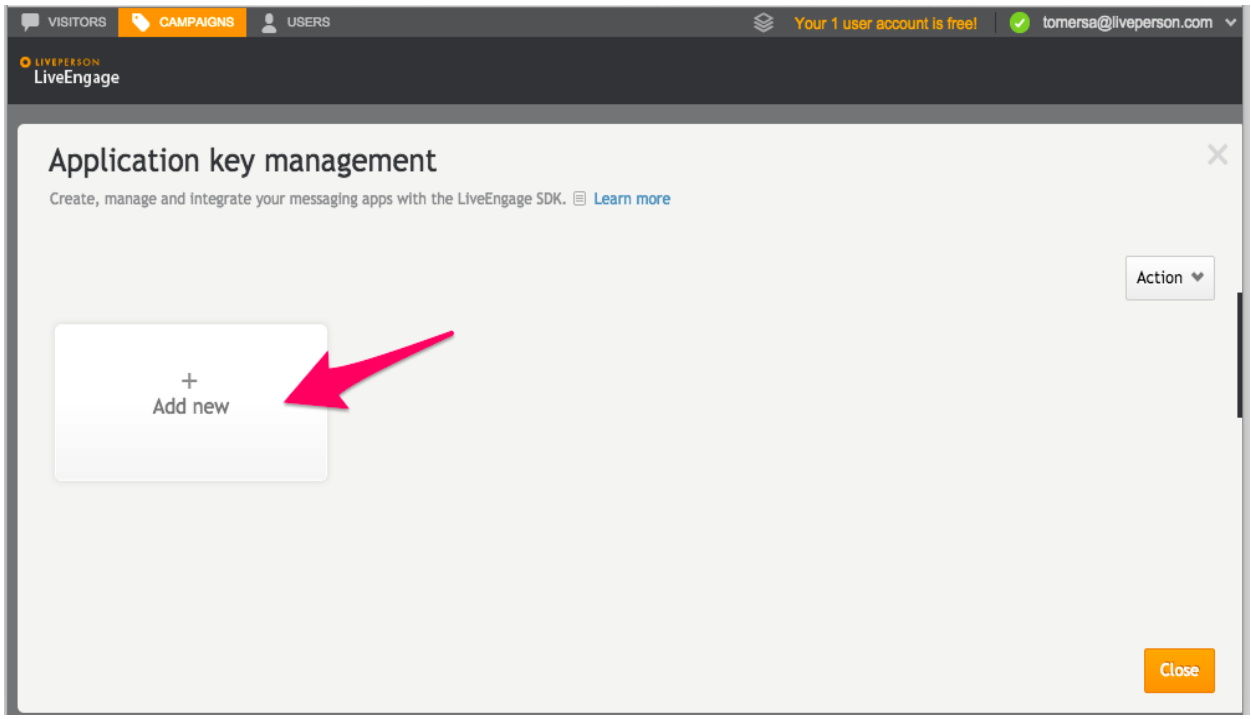



3. Click **Data Sources**, and then select **App**.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.




4. Click **Configure**.

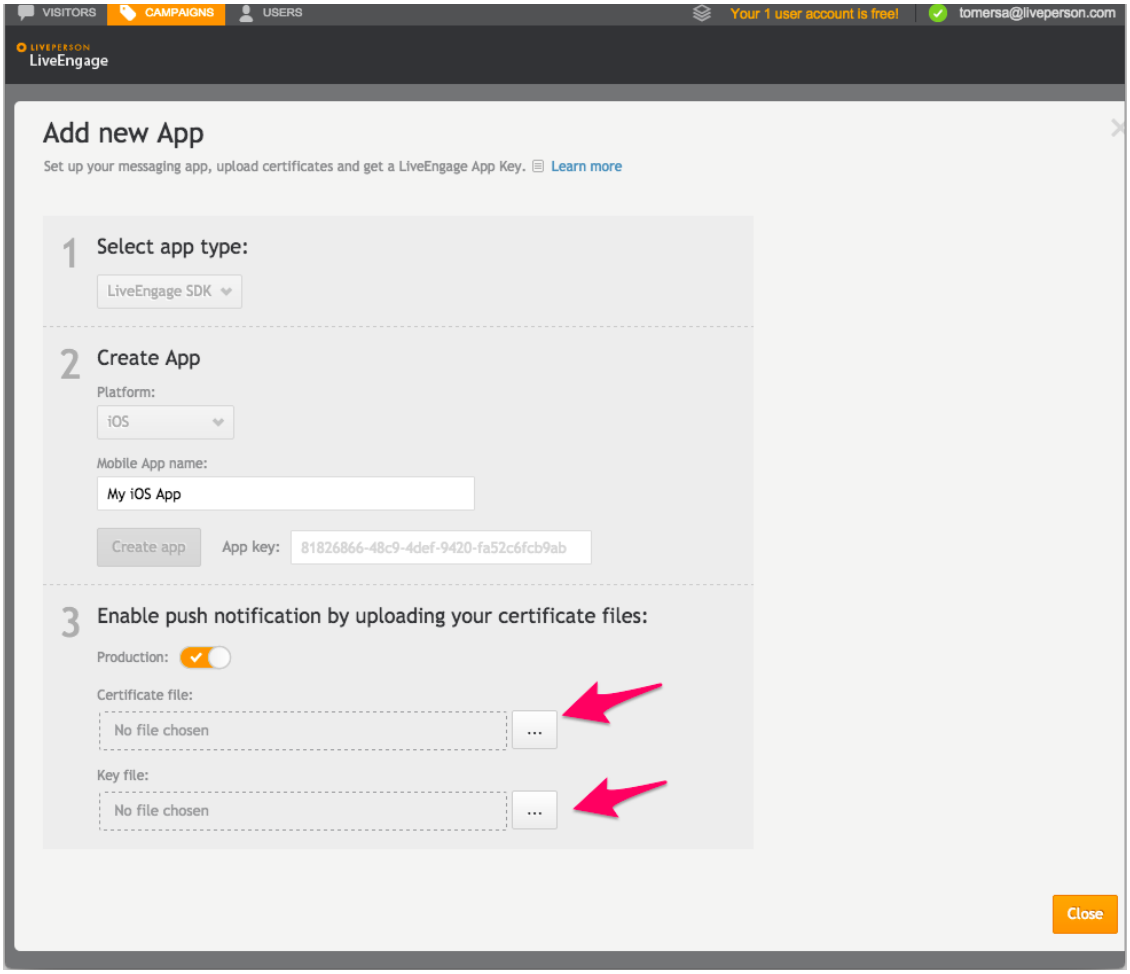



**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

5. Click **Add new** to associate your app with the LiveEngage account.

6. Select your platform as iOS, enter your app's name, and click **Create app**. Then, upload your app certificate and key file in the appropriate locations.

*Note: If you are using a development certificate you should uncheck the Production checkbox and add Dev postfix to the Mobile app name.For example, if your app bundle ID is AppId, your mobile app name should be "AppId-Dev". If you are using a production certificate you should leave the production checkbox checked and insert to the Mobile App name your App bundle ID as it is.*




7. Click Close to complete the process.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

# Open Source List

The following open source code is used within the LiveEngage SDK.  Licensing terms for use of this code require you to mention the list of these sources in the end customer product or documentation.  No additional fees or costs are associated with use of these sources.

| Name | License |
|---|---|
| Reachability | Apple inc |
| SocketRocket | BSD |
| UIRefreshControl+UITableView | MIT |
| TTTAttributedLabel | Apache |
| NSDate+Extension | License |

# Appendix

## Security

Security is a top priority and key for enabling trusted, meaningful engagements.

LivePerson's comprehensive security model and practices were developed based on years of experience in SaaS operations, close relationships with Enterprise customers' security teams, frequent assessments with independent auditors, and active involvement in the security community.

LivePerson has a comprehensive security compliance program to help ensure adherence to internationally recognized standards and exceed market expectations. Among the standards LivePerson complies with are: SSAE16 SOC2, ISO27001, PCI-DSS via Secure Widget, Japan's FISC, SafeHarbor, SOX, and more.

Our applications are developed under a strict and controlled Secure Development Life-Cycle: Developers undergo secure development training, and security architects are involved in all

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.

major projects and influence the design process. Static and Dynamic Code Analysis is an inherent part of the development process and, upon maturity, the application is tested for vulnerabilities by an independent penetration testing vendor. On average, LivePerson undergoes 30 penetration tests each year.

**Copyright notice** – ©2017 LivePerson, Inc. All rights reserved.