

Interactive notifications

Overview

With interactive notification, when a notification arrives, users can take actions without opening the application. When an interactive notification arrives, the device shows action buttons along with the notification message.

Interactive notifications are supported on devices with iOS version 8 and above. If an interactive notification is sent to an iOS device with version earlier than 8, the notification actions are not displayed.

Sending interactive push notification

Prepare the notification and send notification. For more information, see [Sending push notifications](#) ([../../sending-notifications](#)).

You can set a string to indicate the category of the notification with the notification object, under **MobileFirst Operations Console** → **[your application]** → **Push** → **Send Notifications** → **iOS custom settings**. Based on the category value, the notification action buttons are displayed. For example:

The screenshot shows the MobileFirst Operations Console interface. On the left is a sidebar with navigation links: Dashboard, mfp runtime, Applications (1), Push Notifications, Versions (1), iOS (latest), App Settings, Push, Adapters (0), Runtime Settings, Error Log, and Devices. The main content area is titled 'Send Notifications' and has tabs for 'Tags' and 'Push Settings'. The 'Send Notifications' tab is active, showing a text input field for 'Enter the notification text.' Below this is the 'iOS Custom Settings' section, which includes fields for 'Badge', 'Sound', 'Action Key', and 'Category'. The 'Badge' field is empty. The 'Sound' field has a green checkmark. The 'Action Key' field has a green checkmark. The 'Category' field has a green checkmark and is set to 'news'. The 'Type' field is set to 'Default'.

Handling interactive push notifications in Cordova application

To receive interactive notifications, follow these steps:

1. In the main JavaScript, define the registered categories for interactive notification and pass it to device register call `MFPPush.registerDevice`.

```

var options = {
  ios: {
    alert: true,
    badge: true,
    sound: true,
    categories: [{
      //Category identifier, this is used while sending the notificat
      ion.
      id : "poll",

      //Optional array of actions to show the action buttons along wi
      th the message.
      actions: [{
        //Action identifier
        id: "poll_ok",

        //Action title to be displayed as part of the notification
        button.
        title: "OK",

        //Optional mode to run the action in foreground or backgrou
        nd. 1-foreground. 0-background. Default is foreground.
        mode: 1,

        //Optional property to mark the action button in red color.
        Default is false.
        destructive: false,

        //Optional property to set if authentication is required or
        not before running the action.(Screen lock).
        //For foreground, this property is always true.
        authenticationRequired: true
      },
      {
        id: "poll_nok",
        title: "NOK",
        mode: 1,
        destructive: false,
        authenticationRequired: true
      }
    ]],

    //Optional list of actions that is needed to show in the case a
    lert.
    //If it is not specified, then the first four actions will be s
    hown.
    defaultContextActions: ['poll_ok','poll_nok'],

    //Optional list of actions that is needed to show in the notifi
    cation center, lock screen.
    //If it is not specified, then the first two actions will be sh
    own.
    minimalContextActions: ['poll_ok','poll_nok']
  }
}

```

2. Pass the `options` object while registering device for push notifications.

```
MFPPush.registerDevice(options, function(successResponse) {  
    navigator.notification.alert("Successfully registered");  
    enableButtons();  
});
```

Handling interactive push notifications in native iOS application

Follow these steps to receive interactive notifications:

1. Enable the application capability to perform background tasks on receiving the remote notifications.
This step is required if some of the actions are background-enabled.
2. Define registered categories for interactive notifications and pass them as options to `MFPPush.registerDevice`.

```
//define categories for Interactive Push  
let acceptAction = UIMutableUserNotificationAction()  
acceptAction.identifier = "OK"  
acceptAction.title = "OK"  
acceptAction.activationMode = .Foreground  
  
let rejetAction = UIMutableUserNotificationAction()  
rejetAction.identifier = "Cancel"  
rejetAction.title = "Cancel"  
rejetAction.activationMode = .Foreground  
  
let category = UIMutableUserNotificationCategory()  
category.identifier = "poll"  
category.setActions([acceptAction, rejetAction], forContext: .Default)  
  
let categories:Set<UIUserNotificationCategory> = [category]  
  
let options = ["alert":true, "badge":true, "sound":true, "categories": categories]  
  
// Register device  
MFPPush.sharedInstance().registerDevice(options as [NSObject : AnyObject],  
completionHandler: {(response: WLResponse!, error: NSError!) -> Void in
```