ExtremeLocation Android SDK Reference Guide

9036427-00 Rev AA

Published August 2019

Copyright © 2019 Extreme Networks, Inc. All rights reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see:
www.extremenetworks.com/company/legal/trademarks

Open Source Declarations

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at:
www.extremenetworks.com/support/policies/software-licensing

Table of Contents

Preface

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

Conventions

This section discusses the conventions used in this guide.

Text Conventions

The following tables list text conventions that are used throughout this guide.

Table 1: Notice Icons

Icon	Notice Type	Alerts you to...
👉	General Notice	Helpful tips and notices for using the product.
📌	Note	Important features or instructions.
⚠	Caution	Risk of personal injury, system damage, or loss of data.
⚡	Warning	Risk of severe personal injury.
New!	New Content	Displayed next to new content. This is searchable text within the PDF.

Table 2: Text Conventions

Convention	Description
`Screen displays`	This typeface indicates command syntax, or represents information as it appears on the screen.
The words enter and type	When you see the word “enter” in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says “type.”
[Key] names	Key names are written with brackets, such as **[Return]** or **[Esc]**. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press **[Ctrl]**+**[Alt]**+**[Del]**
Words in italicized type	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.

Terminology

When features, functionality, or operation is specific to a switch family, such as ExtremeSwitching, the family name is used. Explanations about features and operations that are the same across all product families simply refer to the product as the switch.

Providing Feedback to Us

Quality is our first concern at Extreme Networks, and we have made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you! We welcome all feedback but especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team, you can do so in two ways:

- Use our short online feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@extremenetworks.com.

Please provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal	Search the GTAC (Global Technical Assistance Center) knowledge base, manage support cases and service contracts, download software, and obtain product licensing, training, and certifications.
The Hub	A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
Call GTAC	For immediate support: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)

- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribing to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form with your information (all fields are required).
3. Select the products for which you would like to receive notifications.

   Note
   You can modify your product selections or unsubscribe at any time.

4. Click **Submit**.

Documentation and Training

To find Extreme Networks product guides, visit our documentation pages at:

Current Product Documentation	www.extremenetworks.com/documentation/
Archived Documentation (for earlier versions and legacy products)	www.extremenetworks.com/support/documentation-archives/
Release Notes	www.extremenetworks.com/support/release-notes
Hardware/Software Compatibility Matrices	https://www.extremenetworks.com/support/compatibility-matrices/
White papers, data sheets, case studies, and other product resources	https://www.extremenetworks.com/resources/

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For more information, visit www.extremenetworks.com/education/.

1 ExtremeLocation Android SDK

This guide describes the ExtremeLocation Android SDK's methods, properties, callbacks and experience response structure. You can use this guide to get an in-depth understanding on how to best utilize the ExtremeLocation SDK in conjunction with the ExtremeLocation cloud.

Step-by-Step Integration

1 Go to https://manage.extremelocation.com, log in with your credentials, and create a new App.

Note
Take note of the AppKey and AppSecret that are generated there for use in your application.

2 Create an application using Android SDK version 21 or later.
3 Add the following `.aar` files from the ExtremeLocation SDK download as modules in your application:
   - `FootmarksSdk.aar`

4 Add the following entries to your dependencies in your application's `build.gradle` in addition to the SDK implementation statement.

```
implementation('com.google.android.gms:play-services-ads:15.0.1')
implementation('com.google.android.gms:play-services-location:15.0.1')
implementation('com.google.code.gson:gson:2.6.1')
implementation 'org.greenrobot:eventbus:3.1.1'
implementation 'com.squareup.okhttp3:okhttp:3.11.0'
```

5 Add the following compile options:

```
compileOptions {
    sourceCompatibility JavaVersion.VERSION_1_8
    targetCompatibility JavaVersion.VERSION_1_8
}
```

6 Check out the example project located in the SDK zip to start scanning for beacons.

If targeting your application for Android M (level 23) or higher you are expected to request permissions on an Activity. ExtremeLocation Android SDK requires one further permission be granted before beacon scanning can start.

```
android.permission.ACCESS_FINE_LOCATION
```

Refer to Google documentation for information on how to request permission https://developer.android.com/training/permissions/requesting.

Note

The ExtremeLocation Android SDK will not function without this permission and may respond with an error upon calling the init function().

Initializing the SDK

You need to have an instance of the `FootmarksAPI` class to interact with the ExtremeLocation SDK. It is obtained by calling a static method to initialize the SDK.

```
mFootmarksAPI = FootmarksAPI.init(context, APP_KEY, APP_SECRET, new
    FootmarksAPI.InitCallback() {
    @Override
    public void onError(FootmarksSdkError footmarksSdkError) {
        Log.w(TAG, String.format("Error %1$s: %2$s",
        footmarksSdkError.getError().toString(), footmarksSdkError.getMessage()));
            switch (footmarksSdkError.getError()) {
                // You can do different actions depending on error type
                case LOCATION_PERMISSION_NOT_ENABLED:
                break;
                case BLUETOOTH_NOT_ON:
                break;
                case BLE_NOT_SUPPORTED:
                break;
                case APPKEY_MISSING:
                break;
                case APPSECRET_MISSING:
                break;
                case LOGIN_ERROR:
                break;
            }
        }
    @Override
    public void onSuccess() {
        // API is initialized");
            // This is a good place to call startScan
          }
});
```

Note

APP_KEY and APP_SECRET can be obtained from https://manage.extremelocation.com. If initialization fails, the `onError` callback will be called with error information and type. If the call succeeds, `onSuccess` will be called. The `FootmarksAPI` instance can then be used to start and stop scanning and to set callbacks for beacon and location related events.

Scanning

The SDK is ready to scan for beacons, Nfc tags, Geofences, and return experiences after the successful call to `init()`. To start scanning, call `startScan()`.

This will display a notification about the application in the system status bar. On latest android versions, 8 and higher, this is the only way to make sure the application can detect beacons in the background.

You can customize the notification by passing a notification to the `startScan` call, or just pass `null` and the SDK will use the default notification.

```
mFootmarksAPI.startScan(notification, new FootmarksAPI.InitCallback() {
          @Override
          public void onError(FootmarksSdkError footmarksSdkError) {
              Log.w(TAG, String.format("Error type:%1$s message:%2$s",
            footmarksSdkError.getError().toString(), footmarksSdkError.getMessage()));
              switch (footmarksSdkError.getError()) {
                  case LOCATION_PERMISSION_NOT_ENABLED:
                      break;
                  case BLUETOOTH_NOT_ON:
                      break;
                  case BLE_NOT_SUPPORTED:
                      break;
              }
}
```

Note
`mFootmarksAPI.stopScan()` call will stop the scanning.

Handling Beacon and Experience Responses

Callback Methods

There are two callbacks you can define on the `FootmarksAPI` class instance. They should be used after getting an instance from the `init()` call. You register for them by calling:

```
mFootmarksAPI.onRangeBeacons(beaconList -> {
// Description: Returns all beacons currently in range of the user.
// Your processing code goes here
})
```

and

```
mFootmarksAPI.onCompleteExperiences((beacon, experienceList) -> {
//Description: Returns one or more Experience objects, depending on how your
//experiences/beacons/zones are setup in the management console.
// Your processing code goes here
});
```

Casting Experience Objects

Java

```
switch(exp.getType()){
case ExperienceTypeImage:
    ImageExp image = ((ImageExp)exp);
    ProgressBar progress(ProgressBar)findViewById(R.id.progress);
    image.presentPictureInImageView((ImageView)
    row.findViewById(R.id.image), progress);
    break;

case ExperienceTypeVideo:
    VideoExp video = ((VideoExp)exp);
    String url = video.videoUrl;
    break;
case ExperienceTypeHtml:
    HtmlExp html = ((HtmlExp)exp);
    html.presentHtmlInWebView(((WebView)findViewById(R.id.webview)));
    break;

case ExperienceTypeUrl:
    UrlExp url = ((UrlExp)exp);
    url.presentUrlInWebView(((WebView)findViewById(R.id.webview)));
    break;

case ExperienceTypeCustom:
    CustomExp custom = ((CustomExp)exp);
    String item = custom.text;
    break;

case ExperienceTypeAlert:
    AlertExp alert = ((AlertExp)exp);
    alert.showAlert();
    break;

default:
    break;

}
```

2 Experience Properties

Dynamic Payload Examples
Send Converted Experience Examples

All experiences inherit from the base class, Experience. Currently, there are six experience types: Video, Image, Html, Url, Alert, and Custom. Experiences are created in the ExtremeLocation UI. The properties for the Experience base class and each subclass are described below. You could use the properties however you like. Depending on how you configure your experiences in the ExtremeLocation UI, you may not populate data for all of the properties. In this case, remember to do proper error checking and just retrieve what attributes makes sense.

Java

```
public ExperienceType type: Enum that indicates the given Experience type
public enum ExperienceType {
    ExperienceTypeCustom("custom"),
    ExperienceTypeVideo("video"),
    ExperienceTypeImage("image"),
    ExperienceTypeAlert("alert"),
    ExperienceTypeHtml("html"),
    ExperienceTypeUrl("url"),
    ExperienceTypeUnknown("");
}
public ExperienceAction action: Enum that instructs how to act on the given experience.
public enum ExperienceAction {
    ExperienceActionPassive("passive”),
    ExperienceActionAutoShow("autoShow"),
    ExperienceActionPrompt("prompt”),
    ExperienceActionUnknown("");
}

public String customDetails: Additional details added to the experience
                that did not fall into any of the standard properties.

public String name: The Experiences name on the management console.

public String notifTitle: This field should contain the text you would like to display
                in a Notification to the user.

public String notifDescription: The notificationDescription is an additional field
                that can be used to better describe what you are notifying the user about.
public boolean showNotif: Indicates whether or not to display a notification
                to the user for the given Experience.
```

VideoExp

VideoExp contains video content that should be made available to the user. This class also provides accessory methods to allow you to easily play the video if you would like.

Java

```
public FMDisplayType displayType: Enum that indicates what mode to display the video in.
public enum FMDisplayType {
Fullscreen("fullscreen"),
Large("large"),
Small("small");
}

public FMVideoProvider contentProvider: Enum that informs you of where the video is
hosted.
public enum FMVideoProvider {
    YouTube("youtube"),
    Vimeo("vimeo"),
    Custom("custom");
}

public String videoURL: A URL that points to the video.
public String videoPlayPictureURL: A URL that points to a picture the user must click
                on for the video to play.  This field may or may not exist
                depending on your app’s style.
```

ImageExp

ImageExp contains an image that should be presented to the user. This class contains a helper method to load the Image data using a background thread and upon completion presents the image in an imageView of your choosing.

Java

```
public String imgURL: A URL that points to the image.

public void presentPictureInImageView(ImageView ImageView, ProgressBar progressBar);
```

HtmlExp

HtmlExp contains an html string that should be presented in a WebView or like UI element. Also, included is a helper load the property htmlString directly into a webView provided.

Java

```
public String htmlString: An html content string that should be displayed.

public void presentHtmlInWebView(WebView webView);
```

UrlExp

UrlExp contains an url that should be presented in a WebView or like UI element. Also, included is a helper load the property url directly into a webView provided.

Java

```
public String url: A url that should be loaded.

public void presentUrlInWebView(WebView webView);
```

AlertExp

An Experience that is intended to simply present a notification to the user. This class contains a helper method, showAlert(). showAlert() parses out the notificationTitle and notificationDescription from the experience and presents a notification in the Android notification bar with this string.

Java

```
public void showAlert();
```

CustomExp

This is a catchall Experience that can be modified to fit your use case in the Footmarks Management Console. If used, the customized data will be returned in the text property;

Java

```
public String text: The data to be used for the custom exp (json, xml, text, etc)
```

Dynamic Payload Examples

To create a dynamic payload for sending to another user/device:

1 Use one of the following helper methods.

```
CustomExp custom = Experience.initCustomContentWithText("{\"url\":\"http:\
\somethings.com\"}");
```

```
HtmlExp html = Experience.initHtmlContentWithText("<div id=\"ad\"
      style=\"width:100%;height:90px;\" >\r\n
      <iframe\r\n
        src=\"http://matthewjamestaylor.com/responsive-ads/ad.html\"\r\n
        border=\"0\"\r\n
        scrolling=\"no\"\r\n
        allowtransparency=\"true\"\r\n
        width=\"100%\"\r\n
        height=\"100%\"\r\n
        style=\"border:0;\">\r\n
      </iframe>\r\n
</div>");
```

```
ImageExp image = Experience.initImageContentWithUrl("https:\\encrypted-
tbn1.gstatic.com/
            images?q=tbn:ANd9GcQTh2cCg6qf3MCmeWekAtnYzVdPEbIqmSjB7ohlv2ThP6sOw9IKZQ");
```

```
UrlExp url = Experience.initUrlContentWithUrl("http:\\engadget.com");
```

```
VideoExp video = Experience.initVideoContentWithProvider("youtube",
      FMDisplayType.Fullscreen,"https://www.youtube.com/watch?v=ysh9vRsiBHM");
```

2 Call one of the methods on that experience to send it over to the server for queueing.

```
custom.sendExperienceToDeviceId(deviceId);
custom.sendExperienceToUsername(username);
```

Send Converted Experience Examples

After a user has received and/or interacted with your Experience payload (or if they didn't), you will want to call one of the following methods to let the API (and your reporting dashboard) know what happened after the user received it.

```
Experience experience = incoming experince;
    experience.sendConvertedExperience(Experience.ConvertedAction.Clicked);

experience.sendConvertedExperience(Experience.ConvertedAction.Clicked,true);
    experience.sendConvertedExperience(Experience.ConvertedAction.Watched, true,
            FMConvertedValueType.Seconds, "30");
    experience.sendConvertedExperienceWithCustomAction("Some custom action taken",
            true, Experience.ConvertedValueType.Custom, "value");
    experience.sendConvertedExperienceWithCustomActionAndValueType("Some custom action
            taken", true, "Some custom value type", "value");

experience.sendConvertedExperienceWithCustomValueType(Experience.ConvertedAction.Shared,
          true, "With", "Facebook friend");
```