بسمه تعالی

پژوهش، طراحی و پیاده سازی خدمات اولویت دار

بسترهای توسعه برنامه های کاربردی

توسعه کتابخانه دریافت و نمایش پیام درون برنامه‌ای

شمارهٔ سند..... Toobatech–۰۳۰۶۰۶–JavaSDK Manual-TR-V۲,۰

تاریخ نگارش.....۶ شهریور ماه ۱۴۰۳

شمارهٔ نگارش.....۳/۰

کارفرما..... پژوهشگاه ارتباطات و فناوری اطلاعات

مجری ..... شرکت تحلیل افزار طوبی تک

آدرس: تهران، ونک، خیابان سئول، خیابان ۱۲ متری اول، پلاک ۱، طبقه ۱، واحد ۱ شرکت تحلیل‌افزار طوبی‌تک

تلفن: ۸۶۰۵۱۷۱۳ (۰۲۱) **فاکس**: ۸۸۰۶۶۷۴۰ (۰۲۱) **سایت اینترنتی**: www.Toobatech.com

حق طبع و نشر

فهرست مطالب

۱ مقدمه ۱

۲ افزودن تنظیمات MBaaS ۱

۳ نصب کتابخانه و پیشنیازها ۱

۳-۱ کتابخانه Mbaas ۲

۳-۲ کتابخانه‌ی mbaas-inappmessaging ۲

۳-۳ کتابخانه‌ی mbaas-inappmessaging-display ۲

۴ لاگ‌های اولیه ۲

۵ لاگ‌های دلخواه جهت ارسال پیام به دسته مشخصی از کاربران ۳

۵-۱ ارسال سن کاربر ۳

۵-۲ ارسال جنسیت کاربر ۳

۶ بازیابی توکن دستگاه ۴

۷ هدایت کاربر به صفحات مختلف با تعامل با پیام ۵

۸ ارسال پیام ۶

۸-۱ ارسال پیام از نوع banner ۶

۸-۱-۱ عنوان پیام ۷

۸-۱-۲ بدنه پیام ۷

۸-۱-۳ تصویر پیام ۷

۸-۱-۴ عمل ضربه بر روی banner ۸

۸-۲ ارسال پیام از نوع image ۸

۸-۲-۱ تصویر پیام ۹

۸-۲-۲ عمل ضربه بر روی image ۹

۸-۳ ارسال پیام از نوع modal ۹

۸-۳-۱ عنوان پیام ۱۰

۸-۳-۲ بدنه پیام ۱۱

۸-۳-۳ تصویر پیام ۱۱

۸-۳-۴ دکمه پیام (اختیاری) ۱۱

۸-۴ ارسال پیام از نوع card ۱۱

۸-۴-۱ عنوان پیام ۱۳

۸-۴-۲ بدنه پیام ۱۳

۸-۴-۳ تصویر پیام ۱۳

۸-۴-۴ دکمه (یا دکمه ها) پیام ۱۳

۹ ارسال داده ۱۳
۱۰ رویدادهای پیش فرض ۱۵
۱۰-۱ رویداد APP_LAUNCH ۱۵
۱۰-۲ رویداد ON_FOREGROUND ۱۶
۱۱ رویدادهای سفارشی ۱۶

۱ مقدمه

در این مستند، کتابخانه Java توسعه داده شده برای دریافت و نمایش پیام درون برنامه‌ای و همچنین ارسال لاگ به Log Server بررسی می‌شود.

این کتابخانه از اندروید ۶ (سطح api ۲۳) تا اندروید ۱۵ (سطح api ۳۶) قابل استفاده است.

۲ افزودن تنظیمات MBaaS

وارد پنل به آدرس https://console.xmbaas.ir/ شوید و یک پروژه ایجاد کنید.

داخل پروژه مشخصات برنامه را وارد کنید .(دقت کنید که نام package که وارد میکنید باید کاملا منطبق با نام package در فایل build.gradle باشد و در غیر این صورت سرویس کار نخواهد کرد.)

فایل mbaas-services.json را دانلود کرده و در پوشه ی assets قرار دهید.

۳ نصب کتابخانه و پیشنیازها

برای اضافه کردن کتابخانه ابتدا باید آدرس repository خدمات MBaaS را به repository های موجود در فایل settings.gradle اضافه کنید.

```
dependencyResolutionManagement {
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
        maven {
            url =
uri("https://repo.xstudioo.ir/repository/mbaas/")
        }
    }
}
```

شکل ۱. افزودن آدرس repository به repositoryهای موجود در فایل settings.gradle

۳-۱ کتابخانه Mbaas

برای استفاده از هر سرویس MBaaS ابتدا باید کتابخانه‌ی MBaaS core را اضافه کنید.

```
implementation("com.toobatech.mbaas:mbaas:۱,۰,۰")
```

شکل ۲. افزودن کتابخانه mbaas به dependencies

این کتابخانه وظایف اساسی MBaaS ازجمله دریافت و مدیریت توکن، ارسال لاگ‌های پیش‌فرض و لاگ‌های کاربر و به‌روز رسانی آن‌ها را برعهده دارد.

۳-۲ کتابخانه‌ی mbaas-inappmessaging

جهت دریافت، مدیریت و ذخیره کردن پیام‌های درون برنامه‌ای باید کتابخانه‌ی mbaas-inappmessaging را اضافه کنید.

```
implementation("com.toobatech.mbaas:inappmessaging:۱,۱,۰")
```

شکل ۳. افزدن کتابخانه mbaas-inappmessaging به dependencies

۳-۳ کتابخانه‌ی mbaas-inappmessaging-display

جهت نمایش مدل‌های مختلف پیام درون برنامه‌ای، کتابخانه‌ی mbaas-inappmessaging -display را اضافه کنید.

```
implementation("com.toobatech.mbaas:inappmessaging-display:۱,۱,۰")
```

شکل ۴. افزدن کتابخانه mbaas-inappmessaging -display به dependencies

پس از افزودن کتابخانه‌های مورد نیاز پروژه را سینک کنید.

٤ لاگ‌های اولیه

با افزودن کتابخانه لاگ‌های زیر ارسال خواهند شد.

- مدل و برند دستگاه
- سیستم‌عامل دستگاه
- Timezone دستگاه
- زبان دستگاه
- نسخه اندروید و سطح API آن
- نام اپراتور (دسترسی READ_PHONE_STATE افزوده شده است)
- نام بسته برنامک و نسخه آن
- سورس نصب برنامه
- زمان مراجعه کاربر به برنامک
- زمان کلیک کاربر بر روی اعلان

۵ لاگ‌های دلخواه جهت ارسال پیام به دسته مشخصی از کاربران

این دسته از لاگ‌ها توسط برنامک تأمین می‌شوند و از آن‌ها برای ارسال پیام به دسته خاصی از کاربران استفاده می‌شود. برای ارسال لاگ‌ها دلخواه خود از تابع setUserProperty استفاده کنید.

۵-۱ ارسال سن کاربر

برای ارسال سن کاربر، تابع setUserProperty را با نام age فرخوانی کنید (برای مثال در هنگام ارسال فرم یا تکمیل پروفایل توسط کاربر). سن باید یک عدد و معتبر باشد. نتیجه یک <Task<String با پیام موفقیت یا عدم موفقیت است.

۵-۲ ارسال جنسیت کاربر

برای ارسال جنسیت کاربر، تابع setUserProperty را با نام gender فرخوانی کنید (برای مثال در هنگام ارسال فرم یا تکمیل پروفایل توسط کاربر). مقادیر قابل قبول برای جنسیت رشته‌های M برای مرد و F برای زن هستند. نتیجه یک <Task<String با پیام موفقیت یا عدم موفقیت است.

```
Button btnSubmit = findViewById(R.id.button_submit);
        btnSubmit.setOnClickListener(v -> {
            TextView tvAge = findViewById(R.id.editText_age);
            String age = tvAge.getText().toString().trim();
            if (!age.isEmpty())
                MBaaS.getInstance()
                        .setUserProperty("age",
tvAge.getText().toString());
        });
        RadioGroup radioGroup = findViewById(R.id.radioGroup);
        radioGroup.setOnCheckedChangeListener((group, checkedId)
-> {
            String gender = null;
            // Check which radio button was clicked
            if (checkedId == R.id.radioButton_male) {
                gender = "M";
            } else if (checkedId == R.id.radioButton_female) {
                gender = "F";
            }
            // If a valid gender is selected, call setUserProperty
            if (gender != null) {
                MBaaS.getInstance().setUserProperty("gender",
gender);
            }
        });
```

شکل ۵. ارسال لاگ‌های اختیاری برای دسته‌بندی کاربران

٦ بازیابی توکن دستگاه

در اجرای اول برنامک، کتابخانه یک توکن برای کاربر ایجاد می‌کند. برای ارسال پیام به یک دستگاه مشخص یا ایجاد گروهی از دستگاه‌ها نیاز است. برای دسترسی به توکن فعلی MBaaS.getInstance().getToken() را صدا بزنید:

```
MBaaS.getInstance().getToken().addOnCompleteListener(task -> {
            if (!task.isSuccessful()) {
                Log.w(TAG, "Fetching registration token failed",
task.getException());
                Toast.makeText(Example.this,
R.string.error_message_token, Toast.LENGTH_SHORT).show();
                return;
            }
            // Get new registration token
            String token = task.getResult();

            Log.d(TAG, token);
            TextView tv = findViewById(R.id.textView_token);
            tv.setText(token);
        });
```

شکل۶. متد دریافت توکن فعلی دستگاه

۷ هدایت کاربر به صفحات مختلف با تعامل با پیام

به صورت پیش‌فرض هنگام کلیک بر روی پیام یا هر یک از دکمه‌ها پیام از بین می‌رود (dismiss می‌شود). جهت تغییر عملکرد پیش‌فرض آن، می‌توانید هنگام ارسال پیام در کنسول، پارامتر action پیام را با یک url (آدرس یک سایت) یا آن را با deepLink یکی از صفحات برنامه ست کنید.

این deepLink باید برای یک Activity در فایل manifest تعریف شده باشد. برای مثال در شکل زیر صفحه پروفایل با deeplink **mbaas://profile** ست شده است.

```
  <activity
            android:name=".Profile"
            android:exported="true"
            android:label="@string/profile">
            <intent-filter>
                <category
android:name="android.intent.category.DEFAULT" />
                <action android:name="OPEN_PROFILE" />
```

```
            </intent-filter>
            <intent-filter>
                <action android:name="android.intent.action.VIEW"
/>
                <category
android:name="android.intent.category.DEFAULT" />
                <category
android:name="android.intent.category.BROWSABLE" />
                <!-- Example deep link: mbaas://profile -->
                <data android:scheme="mbaas"
android:host="profile" />
            </intent-filter>
</activity>
```

شکل ۷. تعریف deeplink در فایل manifest برای رفتن به صفحه‌ای خاص از برنامه

۸ ارسال پیام

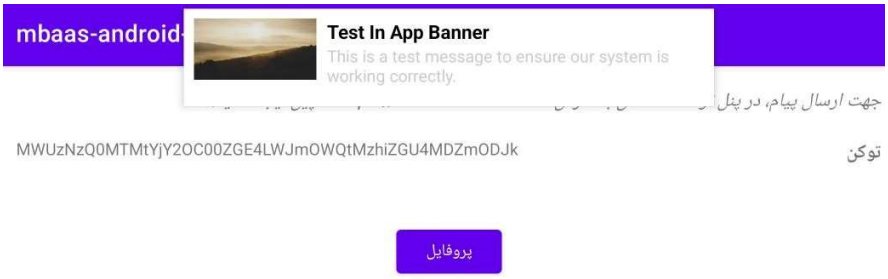
۸-۱ ارسال پیام از نوع banner

برای ارسال پیام درون برنامه‌ای از نوع banner با مشخصات مد نظر است که توسط کاربر قابل مشاهده خواهد بود، می‌توانید فیلدهای banner را به صورت زیر ست کنید.

پیام banner به این صورت می‌باشد:

شکل ۸ نمایش پیام درون برنامه‌ای از نوع banner به صورت عمودی

بنر برای صفحات نمایشی با اندازه‌های مختلف و چرخش دستگاه به صورت responsive طراحی شده است.



شکل۹. نمایش پیام درون برنامه‌ای از نوع banner به صورت افقی

۸-۱-۱ عنوان پیام

برای مشخص کردن عنوان پیام فیلد title را به صورت رشته ست کنید.

عنوان پیام نمایشی باید مقدار title باشد.

۸-۱-۲ بدنه پیام

برای مشخص کردن متن پیام فیلد body را به صورت رشته ست کنید.

متن پیام نمایشی باید مقدار text باشد.

۸-۱-۳ تصویر پیام

فیلد image را با url تصویر ست کنید.

پیام باید با تصویر مشخص شده نمایش داده شود.

۸-۱-٤ عمل ضربه بر روی banner

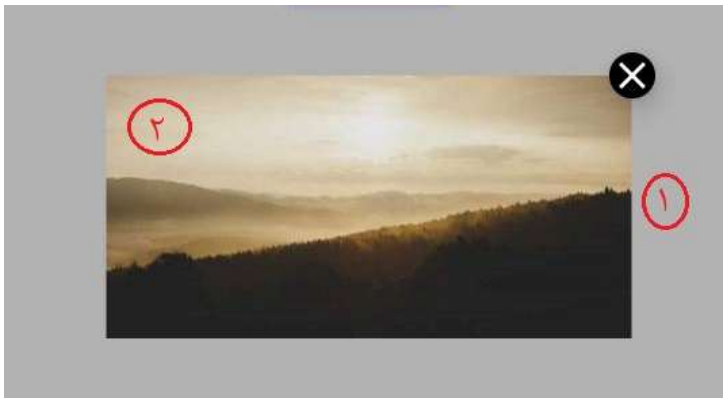
مشخص‌کننده عملی است که با ضربه بر روی banner باید رخ دهد. فیلد actionUrl را با یک url مقداردهی کنید.

با ست کردن actionUrl هنگام ضربه برروی پیام باید url یا deeplink مربوطه باز شود.

۸-۲ ارسال پیام از نوع image

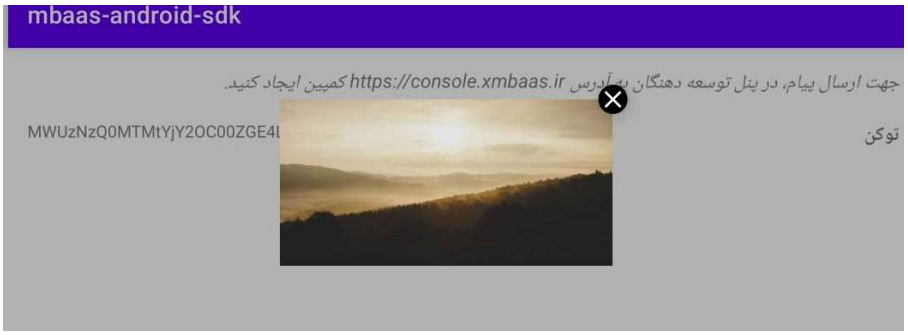
برای ارسال پیام درون برنامه‌ای از نوع image با مشخصات مد نظر است که توسط کاربر قابل مشاهده خواهد بود.برای ارسال پیام با مشخصات مورد نظر خود، می‌توانید فیلدهای image را به صورت زیر ست کنید.

پیام image به این صورت می‌باشد:



شکل ۱۰. نمایش پیام درون برنامه‌ای از نوع image به صورت عمودی

تصویر برای صفحات نمایشی با اندازه‌های مختلف و چرخش دستگاه به صورت responsive طراحی شده است.



شکل ۱۱. نمایش پیام درون برنامه‌ای از نوع image به صورت افقی

۸-۲-۱ تصویر پیام

فیلد image را با url تصویر ست کنید.

پیام باید با تصویر مشخص شده نمایش داده شود.

۸-۲-۲ عمل ضربه بر روی image

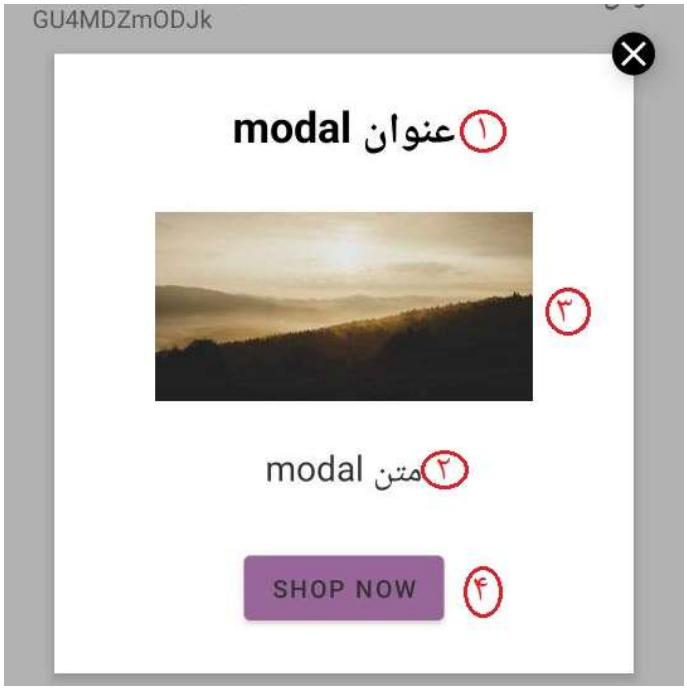
مشخص‌کننده عملی است که با ضربه بر روی image باید رخ دهد. فیلد actionUrl را با یک url مقداردهی کنید.

با ست کردن actionUrl هنگام ضربه برروی پیام باید url یا deeplink مربوطه باز شود.

۸-۳ ارسال پیام از نوع modal

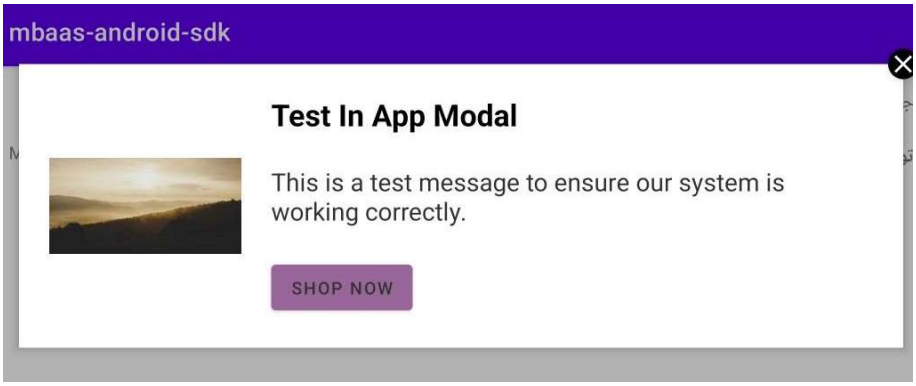
برای ارسال پیام درون برنامه‌ای از نوع modal با مشخصات مد نظر است که توسط کاربر قابل مشاهده خواهد بود. برای ارسال پیام با مشخصات مورد نظر خود، می‌توانید فیلدهای modal را به صورت زیر ست کنید.

پیام modal به این صورت می‌باشد:



شکل۱۲. نمایش پیام درون برنامه‌ای از نوع modal به صورت عمودی

مودال برای صفحات نمایشی با اندازه‌های مختلف و چرخش دستگاه به صورت responsive طراحی شده است.



شکل۱۳. نمایش پیام درون برنامه‌ای از نوع modal به صورت افقی

۱-۳-۸ عنوان پیام

برای مشخص کردن عنوان پیام فیلد title را به صورت رشته ست کنید.

عنوان پیام نمایشی باید مقدار title باشد.

۸-۳-۲ بدنه پیام

برای مشخص کردن متن پیام فیلد body را به صورت رشته ست کنید.

متن پیام نمایشی باید مقدار text باشد.

۸-۳-۳ تصویر پیام

فیلد image را با url تصویر ست کنید.

پیام باید با تصویر مشخص شده نمایش داده شود.

۸-۳-٤ دکمه پیام (اختیاری)

برای مشخص کردن دکمه پیام فیلد button را به صورت رشته ست کنید.

پیام باید با دکمه مشخص شده نمایش داده شود.

با ست کردن actionUrl هنگام ضربه برروی دکمه باید url یا deeplink مربوطه باز شود.

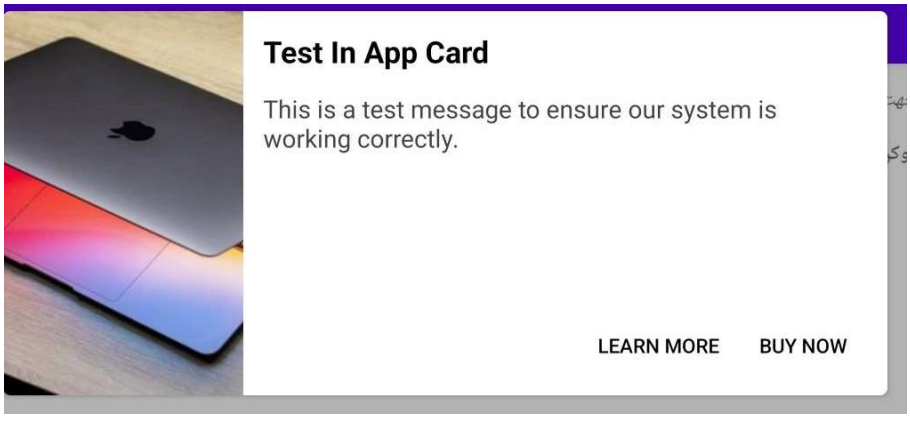
۸-٤ ارسال پیام از نوع card

برای ارسال پیام درون برنامه‌ای از نوع card با مشخصات مد نظر است که توسط کاربر قابل مشاهده خواهد بود، می‌توانید فیلدهای card را به صورت زیر ست کنید.

پیام card به این صورت می‌باشد:

شکل۱۴. نمایش پیام درون برنامه‌ای از نوع card به صورت عمودی

کارت برای صفحات نمایشی با اندازه‌های مختلف و چرخش دستگاه به صورت responsive طراحی شده است.



شکل۱۵. نمایش پیام درون برنامه‌ای از نوع card به صورت افقی

۸-٤-۱ عنوان پیام

برای مشخص کردن عنوان پیام فیلد title را به صورت رشته ست کنید.

عنوان پیام نمایشی باید مقدار title باشد.

۸-٤-۲ بدنه پیام

برای مشخص کردن متن پیام فیلد body را به صورت رشته ست کنید.

متن پیام نمایشی باید مقدار text باشد.

۸-٤-۳ تصویر پیام

فیلد image را با url تصویر ست کنید.

پیام باید با تصویر مشخص شده نمایش داده شود.

۸-٤-٤ دکمه (یا دکمه ها) پیام

در مدل نمایشی کارت دکمه اول اجباری و دکمه دوم اختیاری است. برای مشخص کردن دکمه (یا دکمه ها) پیام فیلد button را به صورت رشته ست کنید.

پیام باید با دکمه (یا دکمه ها) مشخص شده نمایش داده شود.

با ست کردن actionUrl هنگام ضربه برروی دکمه باید url یا deeplink مربوطه باز شود.

۹ ارسال داده

هنگام سفارشی سازی کردن پیام می‌توان به فیلد data و سایر فیلدهای پیام دسترسی داشت. جهت ارسال داده در کنسول و یا در api فیلد data را با کلید-مقدارهای مورد نظر خود ست کنید.

شکل۱۶. نمونه بدنه‌ی ارسال داده در کنسول

```
{
    "api_key": "{{api_key}}",
    "username": "{{username}}",
    "app_name": "{{app_name}}",
    "messaging_mode": "inapp",
    "project_id": "{{project_id}}",
    "content": {
        "messageType": "BANNER",
        "campaignMetadata": {
            "campaignName": "Banner Test",
            "isTestMessage": false
        },
        "title": {
            "text": "App launch 11",
            "hexColor": "#000000"
        },
        "body": {
            "text": "This is a test message to ensure our system is working correctly.",
            "hexColor": "#CCCCCC"
        },
        "imageData": {
            "imageUrl": ""
        },
        "backgroundHexColor": "#FFFFFF",
        "action": {
            "actionUrl": "https://example.com/sale"
        },
        "data": {
            "{{$randomTransactionType}}": "{{$randomBankAccount}}" //your custom data
        },
        "triggeringConditions": [
            {
                "miamTrigger": "APP_LAUNCH"
            },
            {
                "analyticsEvent": {"name": "myCustomEvent"}
```

شکل۱۷. نمونه بدنه‌ی ارسال داده در api

۱۰ رویدادهای پیش فرض

جهت نمایش پیام در پیام درون برنامه‌ای از رویدادها استفاده می‌شود. جهت تعیین رویداد از کنسول و یا API استفاده نمایید. دو رویداد ON_FOREGROUND و APP_LAUNCH رویدادهای پیش‌فرض پیام درون برنامه‌ای هستند.

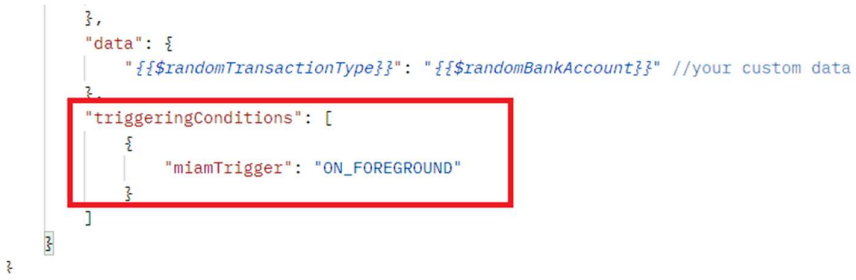
۱۰-۱ رویداد APP_LAUNCH

این رویداد با بازشدن برنامه از حالت بسته (kill) اتفاق می‌افتد.

```
        "actionUrl": "https://console.firebase.google.com/"
    },
    "data": {
        "{{$randomTransactionType}}": "{{$randomBankAccount}}" //your custom data
    },
    "triggeringConditions": [
        {
            "miamTrigger": "APP_LAUNCH"
        }
    ]
}
```

۱۰-۲ رویداد ON_FOREGROUND

این رویداد با بازشدن برنامه از پس‌زمینه و نمایان شدن ظاهر آن اتفاق می‌افتد. هنگام ارسال پیام تست (ارسال به یک کاربر یا توکن خاص) رویدا همواره ON_FOREGROUND خواهد بود.



۱۱ رویدادهای سفارشی

جهت تعریف رویدادهای سفارشی از متد logEvent استفاده کنید.

پارامتر اول این تابع نام رویداد می باشد.

پارامتر دوم آن داده هایی است که در قالب bundle به همراه رویداد به Analytics ارسال خواهد شد .

دریافت پارامترهای رویداد در حال حاضر فعال نیست و در نسخه های بعدی پیاده سازی خواهد شد .

```
import com.toobatech.mbaas.inappmessaging.MBaaSInAppMessaging;

MBaaSInAppMessaging.getInstance(MainActivity.this).logEvent("my_cu
stom_event", new Bundle());
        });
```