Мобильная медиация

Интеграция

10.07.2024

Яндекс

Мобильная медиация. Интеграция. Версия 2.0

Дата подготовки документа: 10.07.2024

Этот документ является составной частью технической документации Яндекса.

Предупреждение об исключительных правах и конфиденциальной информации

Исключительные права на все результаты интеллектуальной деятельности и приравненные к ним средства индивидуализации юридических лиц, товаров, работ, услуг и предприятий, которым предоставляется правовая охрана (интеллектуальную собственность), используемые при разработке, поддержке и эксплуатации службы Мобильная медиация, включая, но не ограничиваясь, программы для ЭВМ, базы данных, изображения, тексты, другие произведения, а также изобретения, полезные модели, товарные знаки, знаки обслуживания, коммерческие обозначения и фирменные наименования, принадлежат ООО «ЯНДЕКС» либо его лицензиарам.

Использование результатов интеллектуальной деятельности и приравненных к ним средств индивидуализации в целях, не связанных с разработкой, поддержкой и эксплуатацией службы Мобильная медиация, не допускается без получения предварительного согласия правообладателя. Настоящий документ содержит конфиденциальную информацию ООО «ЯНДЕКС». Использование конфиденциальной информации в целях, не связанных с разработкой, поддержкой и эксплуатацией службы Мобильная медиация, а равно как и разглашение таковой, не допускается. При этом под разглашением понимается любое действие или бездействие, в результате которых конфиденциальная информация в любой возможной форме (устной, письменной, иной форме, в том числе с использованием технических средств) становится известной третьим лицам без согласия обладателя такой информации либо вопреки трудовому или гражданско-правовому договору.

Отношения ООО «ЯНДЕКС» с лицами, привлекаемыми для разработки, поддержки и эксплуатации службы Мобильная медиация, регулируются законодательством Российской Федерации и заключаемыми в соответствии с ним трудовыми и/или гражданско-правовыми договорами (соглашениями). Нарушение требований об охране результатов интеллектуальной деятельности и приравненных к ним средств индивидуализации, а равно как и конфиденциальной информации, влечет за собой дисциплинарную, гражданско-правовую, административную или уголовную ответственность в соответствии с законодательством Российской Федерации.

Контактная информация

ООО «ЯНДЕКС»

https://www.yandex.ru

Тел.: +7 495 739 7000

Email: pr@yandex-team.ru

Главный офис: 119021, Россия, г. Москва, ул. Льва Толстого, д. 16

Содержание

Подключение плагина....4

Форматы рекламы....5
- Баннерная реклама....5
- Полноэкранная реклама....7
- Реклама с вознаграждением....9

Адаптеры мобильной медиации....11
- AdMob....11
- myTarget....12
- Start.io....12
- UnityAds....13
- AppLovin....14
- IronSource....14
- AdColony....14
- Chartboost....15
- Pangle....15
- Tapjoy....15
- Vungle....16
- Mintegral....16

Подключение Mobile Ads Unity плагина

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Mobile Ads Unity — плагин для игровой платформы Unity3d, включающий поддержку Yandex Mobile Ads SDK.

Примечание:

1. Для работы SDK требуется Target API Level версии 31 и выше.
2. Для загрузки любого вида рекламы необходима версия iOS 12.0 и выше.

Интеграция плагина

Примечание: `yandex-ads-unity-plugin` работает только в окружениях Android и iOS. Работа в редакторе Unity невозможна.

Lite-версия

1. Скачайте каталог `yandex-ads-unity-plugin` и добавьте пакет `yandex-mobileads-lite-.unitypackage`. Вместе с ним будет предложено установить Google resolver. Если в ваш проект уже добавлен Google resolver, уберите галочку.

   Как добавить пакет

   Выберите необходимый плагин (**Assets** → **Import Package** → **Custom Package**) и нажмите кнопку **Import**.
2. С помощью Google resolver установите зависимости: включите **auto-resolve** или выберите в меню пункт **Assets** → **External Dependency Manager** → **Android Resolver** → **Resolve**.
3. Чтобы проверить работу Mobile Ads Unity плагина, воспользуйтесь одним из демонстрационных скриптов в каталоге **samples** репозитория `yandex-ads-unity-plugin`. Скопируйте скрипт в каталог с проектом и добавьте как **Component** в основную камеру.

Подключение мобильной медиации

Автоматическое подключение всех доступных адаптеров

Подключить все доступные адаптеры можно автоматически с помощью общего пакета медиации `yandex-mobileads-mediation`.

1. Скачайте каталог `yandex-ads-unity-plugin` и добавьте пакет `yandex-mobileads-mediation-2.9.0.unitypackage`. Вместе с ним будет предложено установить Google resolver. Если в ваш проект уже добавлен Google resolver, уберите галочку.

   Как добавить пакет

   Выберите необходимый плагин (**Assets** → **Import Package** → **Custom Package**) и нажмите кнопку **Import**.
2. С помощью Google resolver установите зависимости: включите **auto-resolve** или выберите в меню пункт **Assets** → **External Dependency Manager** → **Android Resolver** → **Resolve**.
3. Чтобы проверить работу Mobile Ads Unity плагина, воспользуйтесь одним из демонстрационных скриптов в каталоге **samples** репозитория `yandex-ads-unity-plugin`. Скопируйте скрипт в каталог с проектом и добавьте как **Component** в основную камеру.

Подключение определенного адаптера с помощью соответствующей библиотеки

Если вам не нужно автоматически подключать все доступные адаптеры, воспользуйтесь инструкциями по подключению только необходимых адаптеров.

Понижение Target API Level

Чтобы понизить Target API Level до версии 30, добавьте в `mainTemplate.gradle` и `launcherTemplate.gradle` (если launcherTemplate используется в проекте) явное понижение версии:

```
configurations.all {
    resolutionStrategy {
        force 'androidx.core:core:1.6.0'
```

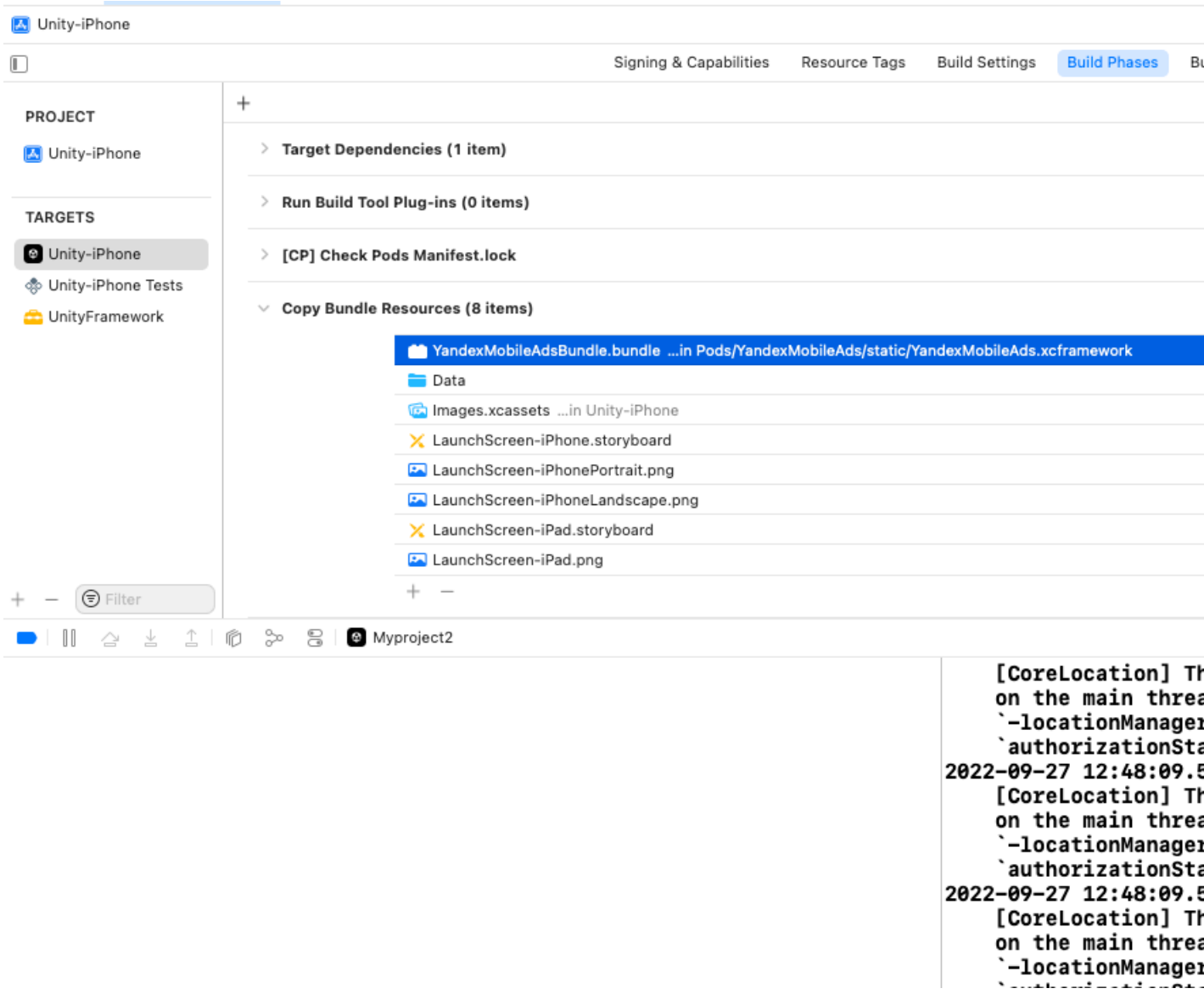
```
        force 'androidx.core:core-ktx:1.6.0'
    }
}
```

Однако, рекомендуется обновление до Target API Level версии 31, так как у Google есть ограничения на выпуск обновлений для приложений с устаревшей версией Target API Level. Подробнее в статье.

Описание ошибок

Полноэкранная реклама Unity Ads не отображается, ошибка «Incorrect fullscreen view»

Во время запуска полноэкранной рекламы на iOS возможна ошибка «Incorrect fullscreen view». При возникновении данной проблемы проверьте, что в настройках **Build Phases**, секции **Copy Bundle Resources** добавлено значение **YandexMobileAdsBundle.bundle**. Если значение отсутствует, добавьте его.



Форматы рекламы

Баннерная реклама

Внимание:

> Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Баннер — это настраиваемое объявление, которое занимает часть экрана и реагирует на нажатие.

Добавление `Banner` в проект

Чтобы отобразить баннер в вашем приложении, создайте объект `Banner` в скрипте (на C#), который прикреплен к `GameObject`.

```
...
using YandexMobileAds;
using YandexMobileAds.Base;
...

public class YandexMobileAdsBannerDemoScript : MonoBehaviour
{
    private Banner banner;
    ...
    private void RequestBanner()
    {
        string adUnitId = "YOUR_adUnitId";

        banner = new Banner(adUnitId, AdSize.BANNER_320x50, AdPosition.BottomCenter);
    }
    ...
}
```

Конструктор `Banner` содержит следующие параметры:

- `AdUnitId` — уникальный идентификатор, который выдается в Партнерском интерфейсе и имеет вид: R-M-XXXXXX-Y.;

  Примечание:

  В интерфейсе Мобильной медиации `AdUnitId` называется ID места.
- `AdSize` — размер баннера, который необходимо показать;
- `AdPosition` — позиция на экране.

Загрузка рекламы

После создания и настройки объекта класса `Banner` необходимо загрузить рекламу. Для загрузки рекламы используйте метод `LoadAd`, принимающий в качестве параметра объект `AdRequest`.

```
...
private void RequestBanner()
{
    ...
    AdRequest request = new AdRequest.Builder().Build();

    banner.LoadAd(request);
    ...
}
...
```

События баннерной рекламы

Чтобы отслеживать события, происходящие в баннерной рекламе, зарегистрируйте делегата для соответствующего `EventHandler`, как показано ниже:

```
...
private void RequestBanner()
{
    ...
        banner.OnAdLoaded += HandleAdLoaded;
        banner.OnAdFailedToLoad += HandleAdFailedToLoad;
        banner.OnReturnedToApplication += HandleReturnedToApplication;
        banner.OnLeftApplication += HandleLeftApplication;
        banner.OnAdClicked += HandleAdClicked;
        banner.OnImpression += HandleImpression;
    ...
}

public void HandleAdLoaded(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleAdLoaded event received");
    banner.Show();
```

```
}

public void HandleAdFailedToLoad(object sender, AdFailureEventArgs args)
{
    MonoBehaviour.print("HandleAdFailedToLoad event received with message: " + args.Message);
}

public void HandleLeftApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleLeftApplication event received");
}

public void HandleReturnedToApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleReturnedToApplication event received");
}

public void HandleAdLeftApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleAdLeftApplication event received");
}

public void HandleAdClicked(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleAdClicked event received");
}

public void HandleImpression(object sender, ImpressionData impressionData)
{
    var data = impressionData == null ? "null" : impressionData.rawData;
    MonoBehaviour.print("HandleImpression event received with data: " + data);
}
```

Очистка рекламы

Когда объект рекламы больше не нужен, его можно удалить. Для этого вызовите метод `Destroy`:

```
banner.Destroy();
```

Полноэкранная реклама

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Полноэкранная реклама (Interstitial) — это настраиваемое объявление, отображаемое на весь экран и реагирующее на нажатие.

Добавление `Interstitial` в проект

Чтобы подключить рекламу, создайте объект `Interstitial` в скрипте (на C#), который прикреплен к `GameObject`.

```
...
using YandexMobileAds;
using YandexMobileAds.Base;
...

public class YandexMobileAdsInterstitialDemoScript : MonoBehaviour
{
    private Interstitial interstitial;
    ...
    private void RequestInterstitial()
    {
        string adUnitId = "YOUR_adUnitId";

        interstitial = new Interstitial(adUnitId);
    }
    ...
}
```

Конструктор `Interstitial` содержит параметр `adUnitId` — уникальный идентификатор, который выдается в Партнерском интерфейсе и имеет вид: R-M-XXXXXX-Y..

Примечание:

В интерфейсе Мобильной медиации `AdUnitId` называется ID места.

Загрузка рекламы

После создания и настройки объекта класса `Interstitial` необходимо загрузить рекламу. Для загрузки рекламы используйте метод `LoadAd`, принимающий в качестве параметра объект `AdRequest`.

```
...
private void RequestInterstitial()
{
    ...
    AdRequest request = new AdRequest.Builder().Build();

    interstitial.LoadAd(request);
    ...
}
...
```

Показ рекламы

После того, как реклама загружена, ее можно показать:

```
...
private void ShowInterstitial()
{
    if (this.interstitial.IsLoaded())
    {
        interstitial.Show();
    }
    else
    {
        Debug.Log("Interstitial is not ready yet");
    }
}
...
```

События полноэкранной рекламы

Чтобы отслеживать события, происходящие в полноэкранной рекламе, зарегистрируйте делегата для соответствующего `EventHandler`, как показано ниже:

```
...
private void RequestInterstitial()
{
    ...
    interstitial.OnInterstitialLoaded += HandleInterstitialLoaded;
    interstitial.OnInterstitialFailedToLoad += HandleInterstitialFailedToLoad;
    interstitial.OnReturnedToApplication += HandleReturnedToApplication;
    interstitial.OnLeftApplication += HandleLeftApplication;
    interstitial.OnAdClicked += HandleAdClicked;
    interstitial.OnInterstitialShown += HandleInterstitialShown;
    interstitial.OnInterstitialDismissed += HandleInterstitialDismissed;
    interstitial.OnImpression += HandleImpression;
    interstitial.OnInterstitialFailedToShow += HandleInterstitialFailedToShow;
    ...
}

public void HandleInterstitialLoaded(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleInterstitialLoaded event received");
}

public void HandleInterstitialFailedToLoad(object sender, AdFailureEventArgs args)
{
    MonoBehaviour.print(
        "HandleInterstitialFailedToLoad event received with message: " + args.Message);
}

public void HandleReturnedToApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleReturnedToApplication event received");
}

public void HandleLeftApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleLeftApplication event received");
}

public void HandleAdClicked(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleAdClicked event received");
}

public void HandleInterstitialShown(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleInterstitialShown event received");
}
```

```
public void HandleInterstitialDismissed(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleInterstitialDismissed event received");
}

public void HandleImpression(object sender, ImpressionData impressionData)
{
    var data = impressionData == null ? "null" : impressionData.rawData;
    MonoBehaviour.print("HandleImpression event received with data: " + data);
}

public void HandleInterstitialFailedToShow(object sender, AdFailureEventArgs args)
{
    MonoBehaviour.print(
        "HandleInterstitialFailedToShow event received with message: " + args.Message);
}
```

Очистка рекламы

Когда объект рекламы больше не нужен, его можно удалить. Для этого вызовите метод `Destroy`:

```
interstitial.Destroy();
```

Реклама с вознаграждением

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Реклама с вознаграждением (Rewarded Ad) — настраиваемое объявление, отображаемое на весь экран. За просмотр такой рекламы пользователь получает вознаграждение.

Добавление `Rewarded Ad` в проект

Чтобы подключить рекламу, создайте объект `RewardedAd` в скрипте (на C#), который прикреплен к `GameObject`.

```
...
using YandexMobileAds;
using YandexMobileAds.Base;
...

public class YandexMobileAdsRewardedAdDemoScript : MonoBehaviour
{
    private RewardedAd rewardedAd;
    ...
    private void RequestRewardedAd()
    {
        string adUnitId = "YOUR_adUnitId";

        rewardedAd = new RewardedAd(adUnitId);
    }
    ...
}
```

Конструктор `RewardedAd` содержит параметр `adUnitId` — уникальный идентификатор, который выдается в Партнерском интерфейсе и имеет вид: R-M-XXXXXX-Y..

Примечание:

В интерфейсе Мобильной медиации `AdUnitId` называется ID места.

Загрузка рекламы

После создания и настройки объекта класса `RewardedAd` необходимо загрузить рекламу. Для загрузки рекламы используйте метод `LoadAd`, принимающий в качестве параметра объект `AdRequest`.

```
...
private void RequestRewardedAd()
{
    ...
    AdRequest request = new AdRequest.Builder().Build();
```

```
    rewardedAd.LoadAd(request);
    ...
}
...
```

Показ рекламы

После того, как реклама загружена, ее можно показать:

```
...
private void ShowRewardedAd()
{
    if (this.rewardedAd.IsLoaded())
    {
        rewardedAd.Show();
    }
    else
    {
        Debug.Log("Rewarded Ad is not ready yet");
    }
}
...
```

События рекламы с вознаграждением

Чтобы отслеживать события, происходящие в рекламе с вознаграждением, зарегистрируйте делегата для соответствующего `EventHandler`, как показано ниже:

```
...
private void RequestRewardedAd()
{
    ...
    rewardedAd.OnRewardedAdLoaded += HandleRewardedAdLoaded;
    rewardedAd.OnRewardedAdFailedToLoad += HandleRewardedAdFailedToLoad;
    rewardedAd.OnReturnedToApplication += HandleReturnedToApplication;
    rewardedAd.OnLeftApplication += HandleLeftApplication;
    rewardedAd.OnAdClicked += HandleAdClicked;
    rewardedAd.OnRewardedAdShown += HandleRewardedAdShown;
    rewardedAd.OnRewardedAdDismissed += HandleRewardedAdDismissed;
    rewardedAd.OnImpression += HandleImpression;
    rewardedAd.OnRewarded += HandleRewarded;
    rewardedAd.OnRewardedAdFailedToShow += HandleRewardedAdFailedToShow;
    ...
}

public void HandleRewardedAdLoaded(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleRewardedAdLoaded event received");
}

public void HandleRewardedAdFailedToLoad(object sender, AdFailureEventArgs args)
{
    MonoBehaviour.print(
        "HandleRewardedAdFailedToLoad event received with message: " + args.Message);
}

public void HandleReturnedToApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleReturnedToApplication event received");
}

public void HandleLeftApplication(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleLeftApplication event received");
}

public void HandleAdClicked(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleAdClicked event received");
}

public void HandleRewardedAdShown(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleRewardedAdShown event received");
}

public void HandleRewardedAdDismissed(object sender, EventArgs args)
{
    MonoBehaviour.print("HandleRewardedAdDismissed event received");
}

public void HandleImpression(object sender, ImpressionData impressionData)
{
    var data = impressionData == null ? "null" : impressionData.rawData;
    MonoBehaviour.print("HandleImpression event received with data: " + data);
}
```

```
public void HandleRewarded(object sender, Reward args)
{
    MonoBehaviour.print("HandleRewarded event received: amout = " + args.amount + ", type = " + args.type);
}

public void HandleRewardedAdFailedToShow(object sender, AdFailureEventArgs args)
{
    MonoBehaviour.print(
        "HandleRewardedAdFailedToShow event received with message: " + args.Message);
}
```

Очистка рекламы

Когда объект рекламы больше не нужен, его можно удалить. Для этого вызовите метод `Destroy`:

```
rewardedAd.Destroy();
```

Адаптеры мобильной медиации

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Мобильная медиация — это платформа автоматического подбора рекламы из нескольких рекламных сетей. Каждая рекламная сеть предложит к показу объявление, а платформа-медиатор выберет из них наиболее выгодное.

Платформа мобильной медиации поддерживает интеграцию адаптеров большинства крупных рекламных сетей, представленных ниже.

Список поддерживаемых рекламных сетей

Рекламная сеть	Баннерная реклама (Banner Ad)	Полноэкранная реклама (Interstitial Ad)	Реклама с вознаграждением (Rewarded Ad)
AdMob	✓	✓	✓
myTarget	✓	✓	✓
Start.io (Android)	✓	✓	✓
UnityAds	✗	✓	✓
AppLovin	✓ (Android)	✓	✓
IronSource	✗	✓	✓
AdColony (Android)	✓	✓	✓
ChartBoost (Android)	✓	✓	✓
Pangle (Android)	✗	✓	✓
Tapjoy (Android)	✓	✓	✓
Vungle (Android)	✓	✓	✓
Mintegral	✓	✓	✓

Подключение AdMob

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-admob-mediation-2.9.0.unitypackage`.
4. Создайте главный файл AndroidManifest.xml через **File** → **Build Settings** → **Android** → **Player Settings** → **Publishing settings** → **Custom Main Manifest** (поставьте галочку).

   Добавьте свой `AdMob ID` в созданный файл AndroidManifest.xml приложения с помощью тега `<meta-data>` с именем com.google.android.gms.ads.APPLICATION_ID (подробнее о том, где найти AdMob ID).

```
<manifest>
    <application>
...
        <meta-data
            android:name="com.google.android.gms.ads.APPLICATION_ID"
            android:value="ca-app-pub-xxxxxxxxxxxxxxxx~yyyyyyyyyy"/>
...
    </application>
</manifest>
```

Подключение myTarget

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-mytarget-mediation-2.9.0.unitypackage`.

Подключение Start.io

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)

- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-startapp-mediation-2.9.0.unitypackage`.

Подключение UnityAds

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-unityads-mediation-2.9.0.unitypackage`.

Описание ошибок

Дублирование классов «Duplicate class com.unity3d.ads.BuildConfig found in modules...»

Если у вас установлен модуль от Unity Ads, билд не соберется и возникнет ошибка дублирования классов:

```
CommandInvokationFailure: Gradle build failed.
/Applications/Unity/Hub/Editor/2021.3.6f1/PlaybackEngines/AndroidPlayer/OpenJDK/bin/java -classpath
 "/Applications/Unity/Hub/Editor/2021.3.6f1/PlaybackEngines/AndroidPlayer/Tools/gradle/lib/gradle-
launcher-6.1.1.jar" org.gradle.launcher.GradleMain "-Dorg.gradle.jvmargs=-Xmx4096m" "assembleRelease"

stderr[
Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':launcher:checkReleaseDuplicateClasses'.
> 1 exception was raised by workers:
  java.lang.RuntimeException: Duplicate class com.unity3d.ads.BuildConfig found in modules jetified-UnityAds-
runtime.jar (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-
ads-4.2.1:)
  Duplicate class com.unity3d.ads.IUnityAdsInitializationListener found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.IUnityAdsLoadListener found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.IUnityAdsShowListener found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.IUnityAdsTokenListener found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAds found in modules jetified-UnityAds-runtime.jar (:UnityAds:) and
 jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAds$UnityAdsInitializationError found in modules jetified-UnityAds-
runtime.jar (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-
ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAds$UnityAdsLoadError found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAds$UnityAdsShowCompletionState found in modules jetified-UnityAds-
runtime.jar (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-
ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAds$UnityAdsShowError found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAdsBaseOptions found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
```

```
  Duplicate class com.unity3d.ads.UnityAdsLoadOptions found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.UnityAdsShowOptions found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.metadata.InAppPurchaseMetaData found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.metadata.MediationMetaData found in modules jetified-UnityAds-runtime.jar
 (:UnityAds:) and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
  Duplicate class com.unity3d.ads.metadata.MetaData found in modules jetified-UnityAds-runtime.jar (:UnityAds:)
 and jetified-com.unity3d.ads.unity-ads-4.2.1-runtime.jar (:com.unity3d.ads.unity-ads-4.2.1:)
...
```

Чтобы исправить ошибку, удалите модуль Unity Ads (**Window** → **Package manager**).

Подключение Applovin

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner) — только для Android
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-applovin-mediation-2.9.0.unitypackage`.

Подключение IronSource

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-ironsource-mediation-2.9.0.unitypackage`.

Подключение AdColony

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-adcolony-mediation-2.9.0.unitypackage`.

Подключение Chartboost

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-chartboost-mediation-2.9.0.unitypackage`.

Подключение Pangle

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-pangle-mediation-2.9.0.unitypackage`.

Подключение Tapjoy

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-tapjoy-mediation-2.9.0.unitypackage`.

Подключение Vungle

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-vungle-mediation-2.9.0.unitypackage`.

Подключение Mintegral

Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится здесь.

Поддерживаемые форматы рекламы

- Баннерная реклама (Banner)
- Полноэкранная реклама (Interstitial)
- Реклама с вознаграждением (Rewarded)

Подключение

1. Настройте медиацию в Партнерском интерфейсе и в кабинете рекламной сети.
2. Импортируйте пакет `yandex-mobileads-lite-2.9.0.unitypackage` в проект.
3. Импортируйте из каталога **mobileads-mediation** пакет `mobileads-mintegral-mediation-2.9.0.unitypackage`.