

Мобильная медиация

Интеграция

10.07.2024

Мобильная медиация. Интеграция. Версия 2.0

Дата подготовки документа: 10.07.2024

Этот документ является составной частью технической документации Яндекс.

© 2008—2024 ООО «ЯНДЕКС». Все права защищены.

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

| | |
|--|-----------|
| Поддержка SKAdNetwork..... | 4 |
| Примечание..... | 4 |
| Подключение Mobile Ads SDK..... | 4 |
| Подключение мобильной медиации..... | 9 |
| Поддержка SKAdNetwork..... | 9 |
| | 20 |
| Форматы рекламы..... | 20 |
| Баннерная реклама..... | 20 |
| Типы баннера..... | 20 |
| Подключение баннера..... | 22 |
| Пример работы с баннером..... | 25 |
| Полноэкранная реклама..... | 26 |
| Реклама с вознаграждением..... | 28 |
| Нативная реклама..... | 29 |
| Процесс подключения..... | 29 |
| Загрузка рекламы..... | 30 |
| Настройка внешнего оформления рекламы..... | 33 |
| Отладка..... | 40 |
| Адаптеры мобильной медиации..... | 41 |
| AdMob..... | 42 |
| myTarget..... | 43 |
| Start.io..... | 44 |
| UnityAds..... | 45 |
| AppLovin..... | 45 |
| IronSource..... | 46 |
| Mintegral..... | 47 |
| AdColony..... | 48 |
| Chartboost..... | 48 |

Ресурсы

Примечание:

Все вызовы Mobile Ads SDK следует выполнять с главного потока.

[Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.

Поддержка SKAdNetwork

Примечание:

SKAdNetwork поддерживается для версии SDK 4.1.2 и выше.

Mobile Ads SDK поддерживает трекинг установок приложений с помощью фреймворка [SKAdNetwork](#). Трекинг установок работает для всех устройств, даже если доступ к IDFA отсутствует.

Чтобы включить функциональность, добавьте идентификаторы поддерживаемых рекламных сетей в файл `Info.plist` приложения.

```
<key>SKAdNetworkItems</key>
  <array>
    <dict>
      <key>SKAdNetworkIdentifier</key>
      <string>zq4921623r.skadnetwork</string>
    </dict>
  </array>
```

Для дополнительной информации ознакомьтесь с разделом [Configuring a Source App](#) документации Apple.

Примечание

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Подключение Mobile Ads SDK



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

1. Для загрузки любого вида рекламы необходима версия iOS 13.0 и выше.
2. Для корректной работы на iOS 14 и выше обратите внимание на [дополнительные шаги](#).



Внимание:

Для работы SDK требуется подключение ресурсов, расположенных в `YandexMobileAdsBundle.bundle` из `YandexMobileAds.framework`. При подключении SDK через `CocoaPods` эти ресурсы подключаются автоматически.

При использовании кастомизированного подключения `YandexMobileAds.framework` обязательно убедитесь в том, что `YandexMobileAds.bundle` копируется в ресурсы проекта.

Библиотека может работать со следующими системами управления зависимостями:

Swift Package Manager

Примечание:

Адаптеры медиации недоступны для подключения через Swift Package Manager. Если вы используете медиацию, рекомендуется интеграция через `CocoaPods`.

Чтобы подключить библиотеку, выполните следующее:

1. В Xcode, в своем проекте добавьте зависимость через **File** → **Add Packages**.
2. Укажите URL репозитория <https://github.com/yandexmobile/yandex-ads-sdk-swift>, в нем находится Swift-пакет.
3. В настройках сборки **Build Settings**, в секции **Linking**, добавьте значение параметра Other Linker Flags = -ObjC.

The screenshot shows the Xcode interface for a project named "SwiftExample". The "Build Settings" tab is selected and highlighted with a red box. The "Linking" section is expanded, also highlighted with a red box, and the "Other Linker Flags" setting is highlighted with a blue bar and a red border. The "Other Linker Flags" setting is currently empty.

| PROJECT | Basic | Customized | All | Combined | Levels |
|--|-------|------------|-----|----------|--------|
| SwiftExample | | | | | |
| TARGETS | | | | | |
| SwiftExample (iOS) | | | | | |
| SwiftExample (macOS) | | | | | |
| Tests iOS | | | | | |
| Tests macOS | | | | | |
| Linking | | | | | |
| Setting | | | | | |
| Bundle Loader | | | | | |
| Compatibility Version | | | | | |
| Current Library Version | | | | | |
| Dead Code Stripping | | | | | |
| Display Mangled Names | | | | | |
| Don't Dead-Strip Inits and Terms | | | | | |
| Dynamic Library Allowable Clients | | | | | |
| Dynamic Library Install Name | | | | | |
| Dynamic Library Install Name Base | | | | | |
| Exported Symbols File | | | | | |
| Generate Position-Dependent Executable | | | | | |
| Initialization Routine | | | | | |
| Link With Standard Libraries | | | | | |
| Mach-O Type | | | | | |
| Order File | | | | | |
| Other Librarian Flags | | | | | |
| Other Linker Flags | | | | | |

4. Проверьте, что таргет слинкован с YandexMobileAdsPackage. Если таргет не слинкован, добавьте связь с библиотекой в секции **Link Binary With Libraries** через значок +.

The screenshot shows the Xcode interface for a project named 'SwiftExample'. The 'Build Settings' tab is selected, and the 'Link Binary With Libraries' section is expanded. The 'Link Binary With Libraries' section is highlighted with a red box. Below this section, there is a table with a header 'Name' and a '+' icon to add new libraries. The '+' icon is also highlighted with a red box. The 'Copy Bundle Resources' section is partially visible at the bottom.

PROJECT

- SwiftExample

TARGETS

- SwiftExample (iOS)
- SwiftExample (ma...)
- Tests iOS
- Tests macOS

Build Settings

- General
- Signing & Capabilities
- Resource Tags
- Info
- Build Settings
- Build Phases

Filter

Dependencies (0 items)

Compile Sources (2 items)

Link Binary With Libraries (0 items)

| Name |
|----------------------------|
| Add frameworks & libraries |

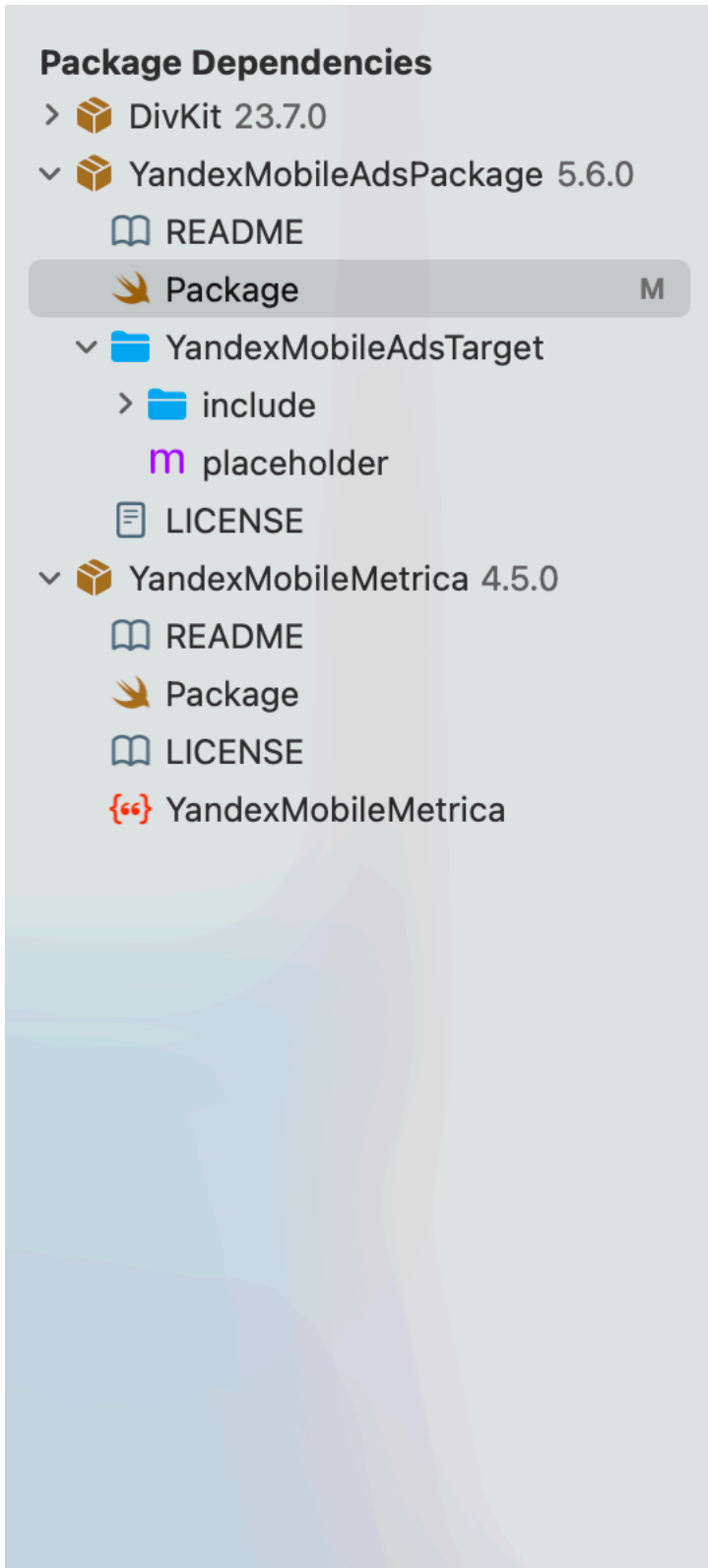
+

Drag to reorder linked binaries

Copy Bundle Resources (Add items)

5. Подключите YandexMobileAdsBundle.bundle из YandexMobileAds. Для этого:

- a. перейдите по ссылке для **YandexMobileAds** в секции **.binaryTarget (Package Dependencies → YandexMobileAdsPackage → кликните на Package.swift → найдите ссылку для YandexMobileAds в секции .binaryTarget)**;

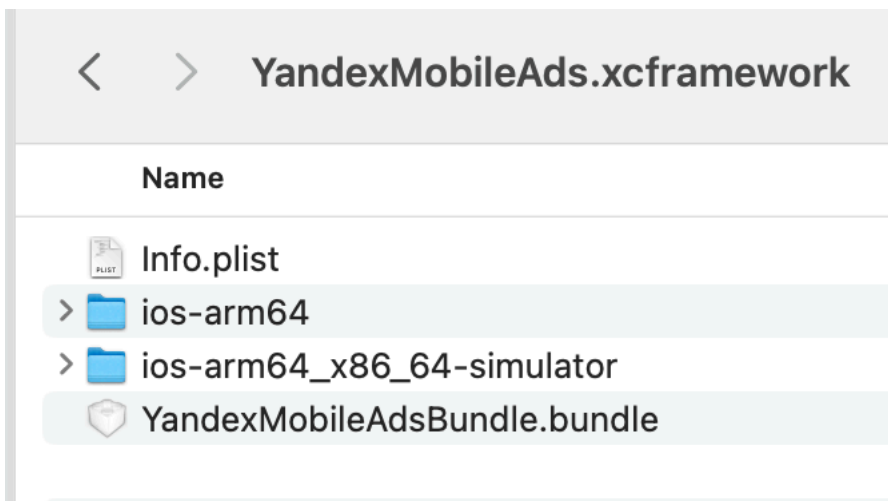


```

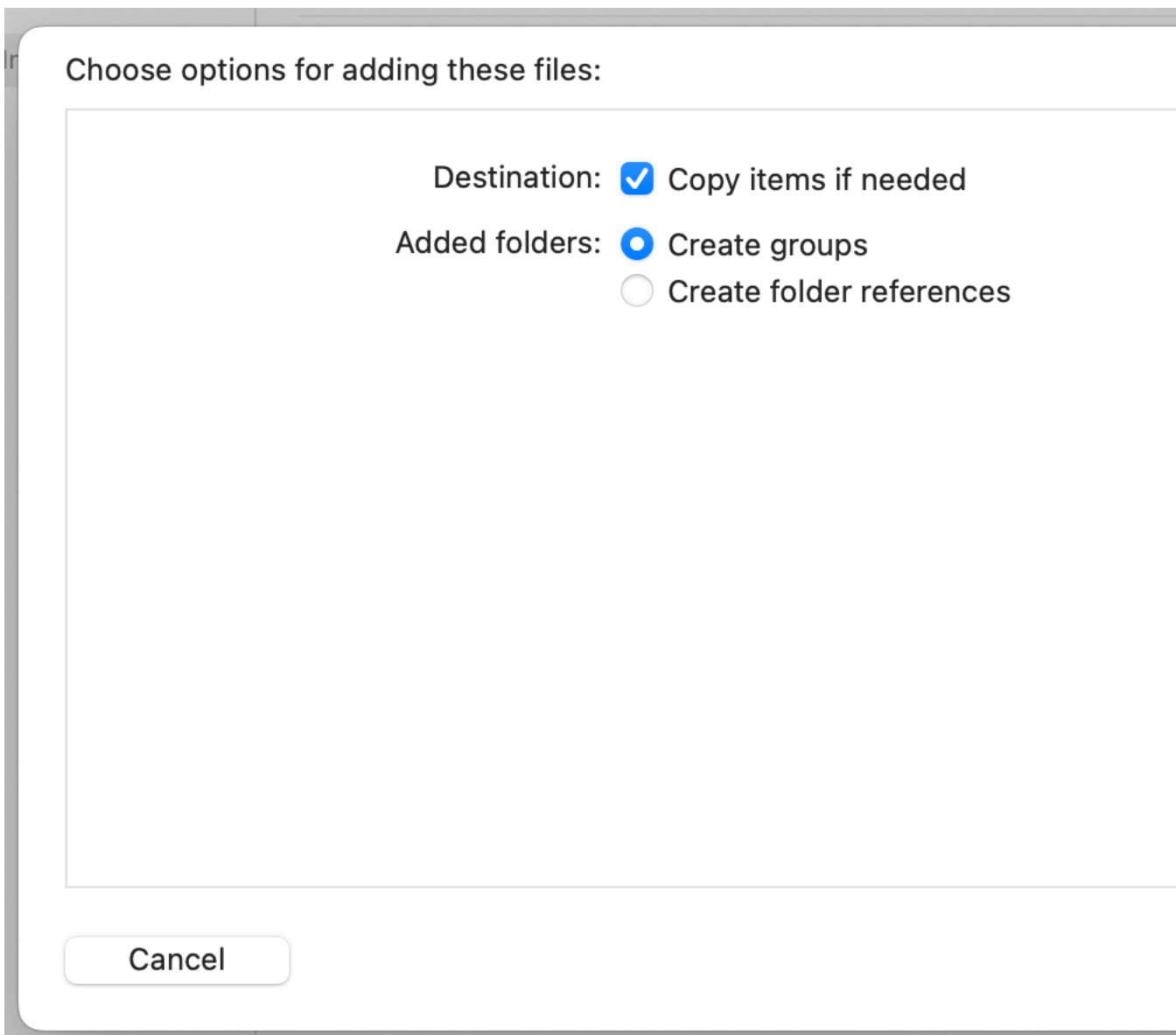
24 dependencies
25     .target(
26     .target(
27     .product
28     .product
29 ],
30 path: "Yande
31 linkerSettin
32     .linkedF
33     .linkedF
34     .linkedF
35     .linkedF
36     .linkedF
37     .linkedF
38     .linkedF
39     .linkedF
40     .linkedF
41     .linkedF
42     .linkedF
43     .linkedF
44     .linkedF
45     .linkedF
46     .linkedF
47     .linkedF
48 ]
49 ),
50 .binaryTarget(
51     name: "Yande
52     url:
53         "https://
54         .0/Yande
55     checksum: "d
56 ),
57 .binaryTarget(
58     name: "Yande
59     url:

```





- b. откройте скачанный архив;



- с. подключите YandexMobileAdsBundle.bundle в Copy Bundle Resource фазу (**Ваш target** → **Build Phases** → **Copy Bundle Resource** → **Кнопка +** → **Add other** → **выберите YandexMobileAdsBundle.bundle** → **выберите пункты Copy items if needed и Create groups**).



Copy Bundle Resources (4 items)

-  YandexMobileAdsBundle.bundle
 -  LaunchScreen.storyboard ...in AdsInstructionSPM/(localization).lproj
 -  Assets.xcassets ...in AdsInstructionSPM
 -  Main.storyboard ...in AdsInstructionSPM/(localization).lproj
- + -

CocoaPods

Библиотека Yandex Mobile Ads SDK адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимости ([пример статического фреймворка](#)):

```
pod 'YandexMobileAds', '5.9.1'  
pod 'YandexMobileAdsInstream', '0.18.0'
```

Подключение мобильной медиации

Есть два способа подключения:

Автоматическое подключение всех доступных адаптеров

Подключить все доступные адаптеры можно автоматически с помощью общей библиотеки медиации YandexMobileAdsMediation. Библиотека адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения. Для этого добавьте в Podfile проекта зависимость:

```
pod 'YandexMobileAdsMediation', '5.9.1'
```

Для каждого адаптера самая свежая совместимая версия будет выбрана автоматически.

Примечание:

Если вам необходим трекинг установок приложений в мобильной медиации, сконфигурируйте Info.plist. Подробнее в разделе [Поддержка SKAdNetwork](#).

Подключение определенного адаптера с помощью соответствующей библиотеки

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

Примечание:

Если вам необходим трекинг установок приложений в мобильной медиации, сконфигурируйте Info.plist. Подробнее в разделе [Поддержка SKAdNetwork](#).

Поддержка SKAdNetwork

Примечание:

SKAdNetwork поддерживается для версии SDK 4.1.2 и выше.

Mobile Ads SDK поддерживает трекинг установок приложений с помощью фреймворка [SKAdNetwork](#). Трекинг установок работает для всех устройств, даже если доступ к IDFA отсутствует.

Чтобы включить функциональность, добавьте идентификаторы поддерживаемых рекламных сетей в файл Info.plist приложения.

```
<key>SKAdNetworkItems</key>
<array>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>zq492l623r.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>633vhxsw4.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>tmhh9296z4.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>vcra2ehyfk.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>zh3b7bxvad.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>xmn954pzmp.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>79w64w269u.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>488r3q3dtq.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>d7g9azk84q.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>nzq8sh4pbs.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>866k9ut3g3.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>2q884k2j68.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>x8jxxk4ff5.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>gfat3222tu.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>pd25vrrwzn.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>lr83yxwka7.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>cp8zw746q7.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>pwdxu55a5a.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>c6k4g5qq8m.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>s39g8k73mm.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>wg4vff78zm.skadnetwork</string>
  </dict>
  <dict>
    <key>SKAdNetworkIdentifier</key>
    <string>g28c52eehv.skadnetwork</string>
  </dict>
</array>
```

```
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>523jb4fst2.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>294199pt4k.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3qy4746246.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>a8cz6cu7e5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ggvn48r87g.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>y755zyxw56.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>qlbq5gtkt8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>mls7yz5dvl.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>67369282zy.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>899vrgt9g8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>mj797d8u6f.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3sh42y64q3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>f38h382jlk.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>24t9a8vw3c.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>mp6xlyr22a.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>x44k69ngh6.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>88k8774x49.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>hs6bdukanm.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>t3b3f7n3x8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>prcb7njmu6.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>c7g47wypnu.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>52f12v3hgk.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9vvzujtq5s.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>m8dbw4sv7c.skadnetwork</string>
```

```
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9g2agbj52.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>m5mvw97r93.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>z5b3gh5ugf.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dd3a75yxkv.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9nlqeag3gk.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>cj5566h2ga.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>h5jmj969g5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dr774724x4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>t7ky8fmwkd.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>fz2k2k5tej.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>u679fj5vs4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>cs644xg564.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9b89h5y424.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>w28pnjg2k4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>2rq3zucswp.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>a7xqa6mtl2.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>g2y4y55b64.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>vc83br9sjg.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>cstr6suwn9.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>eqhxz8m8av.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7k3cvf297u.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>w9q455wk68.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>nu4557a4je.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
```

```
<string>v4nxqhlyqp.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>wzmmz9fp6w.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7fmhfwg9en.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>su67r6k2v3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>yc1nrxr15pm.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7tnzynbdc7.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>l6nv3x923s.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>h8vml93bkz.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>uzqba5354d.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8qieqk9qfv.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>v79kvwwj4g.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>xx9sdjej2w.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>au67k4efj4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>t38b2kh725.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7ug5zh24hu.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>rx5hdcabgc.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5lm9lj6jb7.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>qqp299437r.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>zmvfpc5aq8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9rd848q2bz.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>79pbpufp6p.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dmv22haz9p.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>y5ghdn5j9k.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>n6fk4nfna4.skadnetwork</string>
</dict>
</dict>
```

```
<key>SKAdNetworkIdentifier</key>
<string>7rz58n8ntl.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>v9wtttbfk9.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>n38lu8286q.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>feyaarzu9v.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7fbxrn65az.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>47vhws6wlr.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ejvt5qm6ak.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>b55w3d8y8z.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>v7896pgt74.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5ghnmfs3dh.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>275upjj5gd.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>627r9wr2y5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>kbd757ywx3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>sczv5946wb.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8w3np9182g.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>hb56zgv37p.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9t245vhmpl.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>nrt9jy4kw9.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7953jerfzd.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dn942472g5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6v7lgmsu45.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>cad8qz2s3j.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>n9x2a789qt.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>r26jy69rpl.skadnetwork</string>
</dict>
```

```
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>eh6m2bh4zr.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>jb7bn6koa5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>fkak3gfpt6.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>a2p9lx4jpn.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>97r2b46745.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>22mmun2rn5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>238da6jt44.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>44jx6755aq.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>b9bk5wbcq9.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>k674qkevps.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>t155sbb4fm.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>24zw6aqk47.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4468km3ulz.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>2tdux39lx8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>2u9pt9hc89.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8s468mf13y.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3cgn6rq224.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>glqzh8vgby.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>av6w8kgt66.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>klf5c315u5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>nfqy3847ph.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dticjx1a9i.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ppxm28t8ap.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9wsyqb3ku7.skadnetwork</string>
```

```
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>74b6s63p61.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>xy9t38ct57.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>424m5254lk.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>qu637u8glc.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>f73kdq92p3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>44n7hlldy6.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>kbmxgpxpgc.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5l3tpt7t6e.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ecpz2srf59.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>x5854y7y24.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>f7s53z58qe.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>x8uqf25wch.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>uw77j35x4d.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6964rsfnh4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>gvmwg8q7h5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6yxv74ff7.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>84993kbrcf.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>54nzkqm89y.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>pwa73g5rt2.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>mlmmfzh3r3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9yg77x724h.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>n66cz3y3bx.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>578prtvx9j.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
```



```
<string>4dzt52r2t5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>bvpn9ufa9b.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6qx585k4p6.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>mtkv5xtk9e.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>193v5h6a4m.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>rvh317un93.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>gta9lk7p23.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5tjdwbrq8w.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>r45fhb6rf7.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>32z4fx6l9h.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>e5fvkxwrpn.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8c4e2ghe7u.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>axh5283zss.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3rd42ekr43.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5mv394q32t.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3qcr597p9d.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>v72qych5uu.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ydx93a7ass.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4pfyvq9l8r.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>5a6flpkh64.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4fzdc2evr5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4w7y6s5ca2.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>252b5q8x7y.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>2fnua5tdw4.skadnetwork</string>
</dict>
</dict>
```

```
<key>SKAdNetworkIdentifier</key>
<string>316bd9hu43.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4mn522wn87.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6g9af3uyq4.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6p4ks3rnbw.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>6xzpu9s2p8.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>737z793b9f.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>89z7zv988g.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8m87ys6875.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>8r8llnkz5a.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>bvxub5ada5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>c3frkrj4fj.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>cg4yq2srnc.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dbu4b84rxf.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dkc879ngq3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>dzg6xy7pwj.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>gta8lk7p23.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>hdw39hrw9y.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>hjevpa356n.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>krvm3zuq6h.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ln5gz23vtd.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>ludvb6z3bs.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>m297p6643m.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>p78axw29g.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>pu4na253f3.skadnetwork</string>
</dict>
```

```
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>s69wq72ugq.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>t6d3zquu66.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>vutu7akeur.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>x2jnk7ly8j.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>x5l83yy675.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>y45688jllp.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>yrqpx2mcb.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>z4gj7hsk7h.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>wzmmZ9fp6w.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4pfyvq9L8r.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>V72QYCH5UU.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>2U9PT9HC89.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>3RD42EKR43.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4FZDC2EVR5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7953JERFZD.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>7UG5ZH24HU.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9RD848Q2BZ.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>9T245VHMPL.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>CJ5566H2GA.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>F38H382JLK.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>KLF5C3L5U5.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>M8DBW4SV7C.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>W9Q455WK68.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>WZMMZ9FP6W.skadnetwork</string>
```

```
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>XY9T38CT57.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>YCLNXRL5PM.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>4PFYVQ9L8R.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>TL55SBB4FM.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>BD757YWX3.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>33r6p7g8nc.skadnetwork</string>
</dict>
<dict>
  <key>SKAdNetworkIdentifier</key>
  <string>g69uk9uh2b.skadnetwork</string>
</dict>
</array>
```

Для дополнительной информации ознакомьтесь с разделом [Configuring a Source App](#) документации Apple.

Ознакомьтесь с [примерами использования SDK](#).

Форматы рекламы

Баннерная реклама



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

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

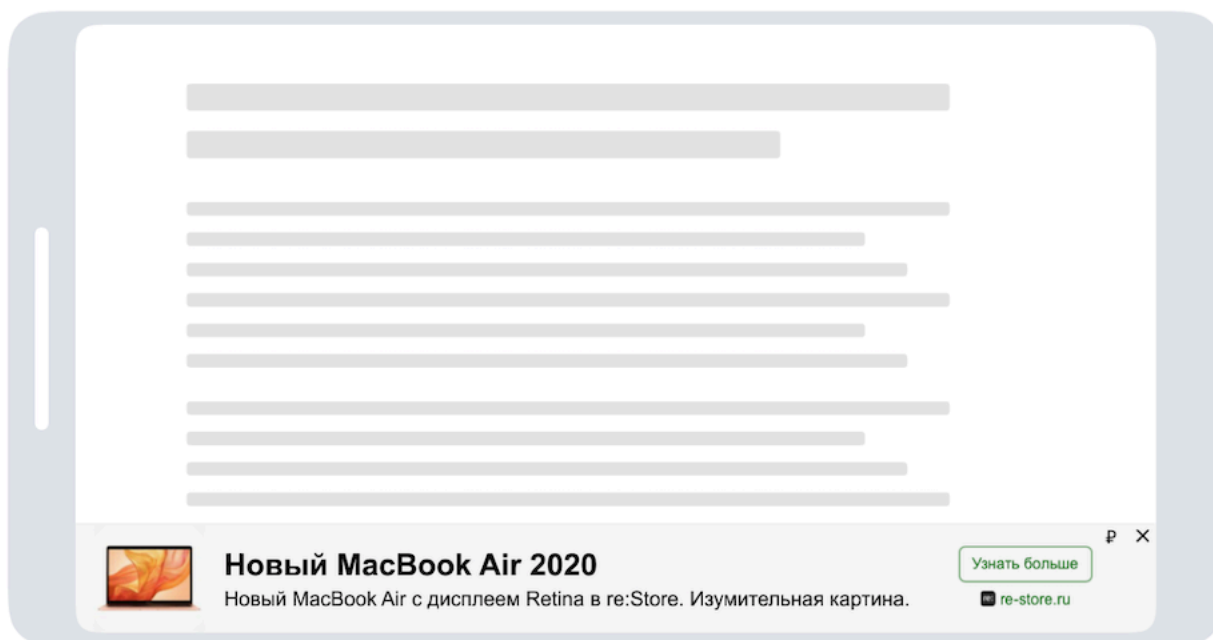
Типы баннера

Sticky баннер

Особенности:

1. Используется заданная ширина баннера. Высота подбирается автоматически.
2. Ширина баннера задается с помощью метода `+stickySizeWithContainerWidth`.
3. Высота баннера не должна превышать 15% высоты устройства и не должна быть меньше 50 dp.

Примеры отображения баннера:

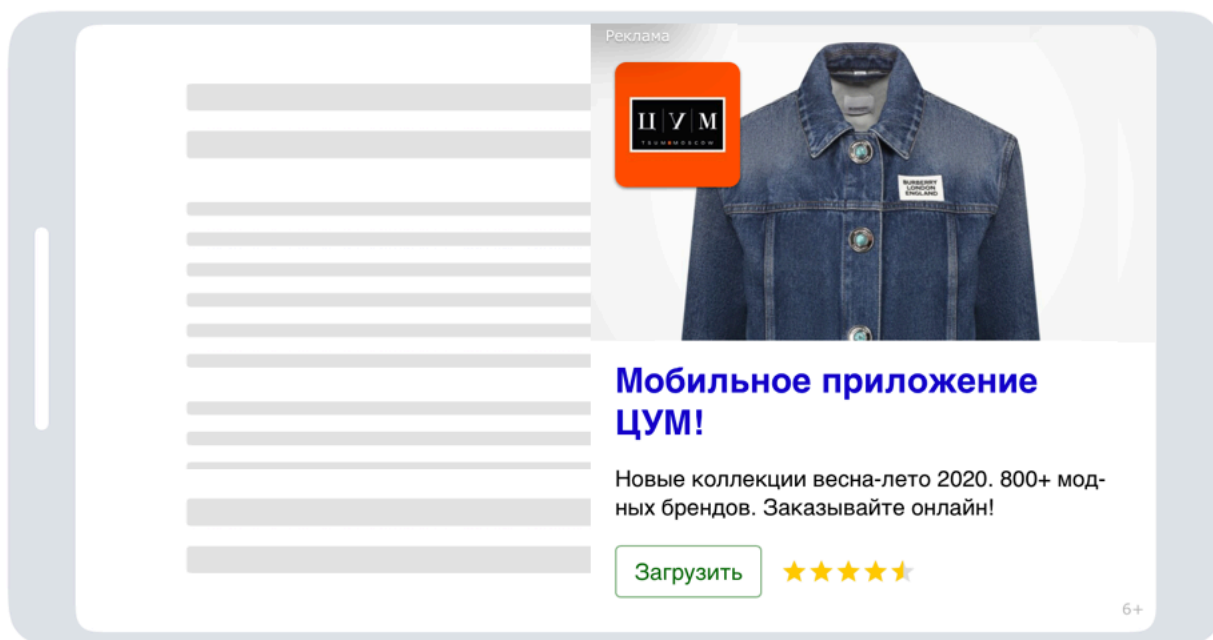


Flex баннер

Особенности:

1. Баннер заполняет весь блок, используя заданные максимальные размеры.
2. Ширина и высота баннера задается с помощью метода `+flexibleSizeWithCGSize:`.
3. SDK накладывает `NSLayoutConstraint` на высоту баннера.

Примеры отображения баннера:



Подключение баннера

Чтобы подключить баннер:

1. Добавьте импорт:

Swift

```
import YandexMobileAds
```

Objective-C

```
#import <YandexMobileAds/YandexMobileAds.h>
```

- Создайте `@property`, где будет храниться ссылка на баннер:

Swift

```
var adView: YMAAView!
```

Objective-C

```
@property (nonatomic, strong) YMAAView *adView;
```

- Создайте баннер:

Sticky баннер

Чтобы задать ширину баннера, вызовите метод [+stickySizeWithContainerWidth:](#).

Swift

```
let adSize = YMAASize.stickySize(withContainerWidth: width)
let adView = YMAAView(adUnitID: "", adSize: adSize)
adView.delegate = self
```

Objective-C

```
YMAASize adSize = [YMAASize stickySizeWithContainerWidth:width];
YMAAView *adView = [[YMAAView alloc] initWithAdUnitID:<AdUnitID> adSize:adSize];
adView.delegate = self;
```

Flex баннер

Чтобы задать ширину и высоту баннера, вызовите метод [+flexibleSizeWithCGSize:](#).

Swift

```
let adSize = YMAASize.flexibleSize(with: size)
let adView = YMAAView(adUnitID: "", adSize: adSize)
adView.delegate = self
```

Objective-C

```
YMAASize adSize = [YMAASize flexibleSizeWithCGSize:size];
YMAAView *adView = [[YMAAView alloc] initWithAdUnitID:<AdUnitID> adSize:adSize];
adView.delegate = self;
```

Ограничение: Требования к размерам баннера при отображении видеорекламы

Минимальный размер баннера, в котором поддерживается воспроизведение видео: 300x160 или 160x300.

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

При этом `self` должен удовлетворять протоколу [YMAAViewDelegate](#). В случае если делегат реализует метод [-viewControllerForPresentingModalView](#), ссылки будут открываться во встроенном браузере внутри приложения. Иначе ссылки будут открываться в браузере, установленном на устройстве.

Чтобы определить причины, по которым реклама работает некорректно, используйте метод [-adViewDidFailLoading:error:](#).

Описание ошибок доступно в разделе [YMAAErrorCode](#).

- Разместите баннер. Вы можете разместить баннер двумя способами:

- С помощью `autolayout constraints`.

Добавьте баннер в `UIView`. Затем добавьте `autolayout constraints`, чтобы баннер отображался в нужном месте.

Swift

```
view.addSubview(adView)
adView.translatesAutoresizingMaskIntoConstraints = false
var adViewConstraints = [
    adView.leadingAnchor.constraint(equalTo: adView.superview!.leadingAnchor),
    adView.trailingAnchor.constraint(equalTo: adView.superview!.trailingAnchor)
]
let bottomDistance: CGFloat = 8
if #available(iOS 11.0, *) {
    adViewConstraints.append(
```

```

        adView.bottomAnchor.constraint(equalTo: view.safeAreaLayoutGuide.bottomAnchor, constant:
        bottomDistance)
    )
} else {
    adViewConstraints.append(
        adView.bottomAnchor.constraint(equalTo: view.bottomAnchor, constant: bottomDistance)
    )
}
NSLayoutConstraint.activate(adViewConstraints)

```

Objective-C

```

UIView *adView = self.adView;
[self.view addSubview:adView];
adView.translatesAutoresizingMaskIntoConstraints = NO;

NSMutableArray *adViewConstraints = [NSMutableArray arrayWithArray:@[
    [adView.leadingAnchor constraintEqualToAnchor:adView.superview.leadingAnchor],
    [adView.trailingAnchor constraintEqualToAnchor:adView.superview.trailingAnchor]
]];
int bottomDistance = 8;
if (@available(iOS 11.0, *)) {
    UILayoutGuide *guide = self.view.safeAreaLayoutGuide;
    [adViewConstraints addObject:[adView.bottomAnchor constraintEqualToAnchor:guide.bottomAnchor
                                constant:bottomDistance]];
} else {
    [adViewConstraints addObject:[adView.bottomAnchor
                                constraintEqualToAnchor:adView.superview.bottomAnchor
                                constant:bottomDistance]];
}
[NSLayoutConstraint activateConstraints:adViewConstraints];

```

- С помощью следующих методов:

Swift

```

displayAtTop(in:)
displayAtBottom(in:)

```

Objective-C

```

-displayAtTopInView;;
-displayAtBottomInView;;

```

В обоих случаях баннеры центрируются по горизонтали.

5. Загрузите баннер. Опционально, с помощью класса [YMAAdRequest](#), можно передать данные для таргетирования.

Swift

```

func loadAd(with request: YMAAdRequest?)

```

Objective-C

```

- (void)loadAdWithRequest:(YMAAdRequest *)request;

```

Когда баннер загрузится, делегат получит об этом уведомление.

6. Опционально можно включить логирование с помощью метода [+enableLogging](#). Если показ рекламы не был засчитан, в консоли появится сообщение.
7. Опционально можно настроить получение уведомлений об окончании проигрывания видео в баннерной рекламе.

Пример реализации

Swift

```

// Получение экземпляра YMAVideoController с помощью videoController.
let videoController = adView.videoController

// Установка делегата, который реализует протокол YMAVideoDelegate.
videoController.delegate = self

// Реализация метода протокола YMAVideoDelegate.
// MARK: - YMAVideoDelegate
func videoControllerDidFinishPlayingVideo(_ videoController: YMAVideoController) {
    print("Video complete");
}

```

Objective-C

```

// Получение экземпляра YMAVideoController с помощью videoController.

```



```

YMAVideoController *viewController = self.adView.videoController;

// Установка делегата, который реализует протокол YMAVideoDelegate.
viewController.delegate = self;

// Реализация метода протокола YMAVideoDelegate.
#pragma mark - YMAVideoDelegate
- (void)videoControllerDidFinishPlayingVideo:(YMAVideoController *)viewController
{
    NSLog(@"%@@", @"Video complete");
}

```

Чтобы посмотреть, как баннерная реклама будет отображаться в приложении, используйте демонстрационный AdUnitID:

```

let yandexAdUnitID = "demo-banner-yandex"
let adMobAdUnitID = "demo-banner-admob"
let mintegralAdUnitID = "demo-banner-mintegral"
let myTargetAdUnitID = "demo-banner-mytarget"
let startAppAdUnitID = "demo-banner-startapp"

```

Понятия, связанные с данным

[Классы и протоколы для работы с баннерной рекламой](#)

[Отслеживание работы рекламы](#)

Узнайте больше

[Пример рекламы](#)

Пример работы с баннером

Следующий код демонстрирует создание и настройку объекта AdView, регистрацию слушателя, а также загрузку баннера.

Swift

```

override func viewDidLoad() {
    super.viewDidLoad()

    // Задание ширины рекламы
    let adSize = YMAAdSize.stickySize(withContainerWidth: containerView())

    // Создание экземпляра adView
    adView = YMAAdView(adUnitID: "R-M-XXXX-YY", adSize: adSize)
    adView.delegate = self
    addAdView()

    // Загрузка рекламы
    adView.loadAd()
}

private func addAdView() {
    view.addSubview(adView)
    adView.translatesAutoresizingMaskIntoConstraints = false
    var adViewConstraints = [
        adView.leadingAnchor.constraint(equalTo: adView.superview!.leadingAnchor),
        adView.trailingAnchor.constraint(equalTo: adView.superview!.trailingAnchor)
    ]
    let bottomDistance: CGFloat = 8
    if #available(iOS 11.0, *) {
        adViewConstraints.append(
            adView.bottomAnchor.constraint(equalTo: view.safeAreaLayoutGuide.bottomAnchor, constant:
            bottomDistance)
        )
    } else {
        adViewConstraints.append(
            adView.bottomAnchor.constraint(equalTo: view.bottomAnchor, constant: bottomDistance)
        )
    }
    NSLayoutConstraint.activate(adViewConstraints)
}

private func containerView() -> CGFloat {
    var containerView = view.frame.width
    if #available(iOS 11, *) {
        containerView = view.frame.inset(by: view.safeAreaInsets).width
    }
    return containerView
}

```

Objective-C

```

- (void)viewDidLoad
{
    [super viewDidLoad];
}

```

```

// Задание ширины рекламы
YMAAdSize *adSize =
    [YMAAdSize stickySizeWithContainerWidth:[self containerWidth]];

// Создание экземпляра adView
self.adView = [[YMAAdView alloc] initWithAdUnitID:@"R-M-XXXX-YY" adSize:adSize];
self.adView.delegate = self;
[self addAdView];

// Загрузка рекламы
[self.adView loadAd];
}

- (void)addAdView
{
    UIView *adView = self.adView;
    [self.view addSubview:adView];
    adView.translatesAutoresizingMaskIntoConstraints = NO;

    NSMutableArray *adViewConstraints = [NSMutableArray arrayWithArray:@[
        [adView.leadingAnchor constraintEqualToAnchor:adView.superview.leadingAnchor],
        [adView.trailingAnchor constraintEqualToAnchor:adView.superview.trailingAnchor]
    ]];
    int bottomDistance = 8;
    if (@available(iOS 11.0, *)) {
        UILayoutGuide *guide = self.view.safeAreaLayoutGuide;
        [adViewConstraints addObject:[adView.bottomAnchor constraintEqualToAnchor:guide.bottomAnchor
                                     constant:bottomDistance]];
    } else {
        [adViewConstraints addObject:[adView.bottomAnchor constraintEqualToAnchor:adView.superview.bottomAnchor
                                     constant:bottomDistance]];
    }
    [NSLayoutConstraint activateConstraints:adViewConstraints];
}

- (CGFloat)containerWidth
{
    CGFloat containerWidth = self.view.frame.size.width;
    if (@available(iOS 11.0, *)) {
        containerWidth =
            UIEdgeInsetsInsetRect(self.view.frame, self.view.safeAreaInsets).size.width;
    }
    return containerWidth;
}

```

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.

Чтобы подключить рекламу:

1. Добавьте импорт:

Swift

```
import YandexMobileAds
```

Objective-C

```
#import <YandexMobileAds/YandexMobileAds.h>
```

2. Создайте `@property`, где будет храниться ссылка на баннер:

Swift

```
var interstitialAd: YMAInterstitialAd!
```

Objective-C

```
@property (nonatomic, strong) YMAInterstitialAd *interstitialAd;
```

3. Произведите инициализацию и предзагрузку рекламы. Это необходимо выполнить после отображения интерфейса приложения.

Опционально, с помощью класса [YMAAdRequest](#), можно передать данные для таргетирования. Пример использования класса см. в разделе [Полноэкранная реклама](#).

Swift

```
interstitialAd = YMAInterstitialAd(adUnitID: "<AdUnitID>")
interstitialAd.delegate = self
interstitialAd.load()
```

Objective-C

```
self.interstitialAd = [[YMAInterstitialAd alloc] initWithAdUnitID:<ваш уникальный AdUnitID>];
self.interstitialAd.delegate = self;
[self.interstitialAd load];
```

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

При этом self должен удовлетворять протоколу [YMAInterstitialAdDelegate](#). Производить предзагрузку рекламы необходимо в той же ориентации, в которой она будет показана (иначе реклама не будет показана из-за несоответствия размеров экрана и баннера).

4. Начните отображение объявления, используя данный метод:

Swift

```
func interstitialAdDidLoad(_ interstitialAd: YMAInterstitialAd) {
    interstitialAd.present(from: self)
}
```

Objective-C

```
-(void)interstitialAdDidLoad:(YMAInterstitialAd *)interstitialAd
{
    [interstitialAd presentFromViewController:self];
}
```

5. Опционально можно включить логирование с помощью метода [+enableLogging](#). Если показ рекламы не был засчитан, в консоли появится сообщение.

Чтобы определить причины, по которым реклама работает некорректно, используйте методы

Swift

```
func interstitialAdDidFail(toLoad interstitialAd: YMAInterstitialAd, error: Error)
func interstitialAdDidFail(toPresent interstitialAd: YMAInterstitialAd, error: Error)
```

Objective-C

```
-(void)interstitialAdDidFailToLoad:(YMAInterstitialAd *)interstitialAd error:(NSError *)error;
-(void)interstitialAdDidFailToPresent:(YMAInterstitialAd *)interstitialAd error:(NSError *)error;
```

Описание ошибок доступно в разделе [YMAAdErrorCode](#).

Чтобы посмотреть, как реклама будет отображаться в приложении, используйте демонстрационный AdUnitID:

```
let yandexAdUnitID = "demo-interstitial-yandex"
let adMobAdUnitID = "demo-interstitial-admob"
let appLovinAdUnitID = "demo-interstitial-applovin"
let ironSourceAdUnitID = "demo-interstitial-ironsource"
let mintegralAdUnitID = "demo-interstitial-mintegral"
let myTargetAdUnitID = "demo-interstitial-mytargеt"
let startAppAdUnitID = "demo-interstitial-startapp"
let unityAdsAdUnitID = "demo-interstitial-unityads"
```

Понятия, связанные с данным

[Классы и протоколы для работы с полноэкранной рекламой](#)

[Отслеживание работы рекламы](#)

Узнайте больше

[Пример рекламы](#)

Процесс подключения рекламы 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.

Чтобы подключить рекламу:

1. Добавьте импорт:

Swift

```
import YandexMobileAds
```

Objective-C

```
#import <YandexMobileAds/YandexMobileAds.h>
```

2. Создайте @property, где будет храниться ссылка на рекламное объявление:

Swift

```
var rewardedAd: YMARewardedAd!
```

Objective-C

```
@property (nonatomic, strong) YMARewardedAd *rewardedAd;
```

3. Произведите инициализацию рекламы:

Swift

```
rewardedAd = YMARewardedAd(adUnitID: "<AdUnitID>")  
rewardedAd.delegate = self
```

Objective-C

```
self.rewardedAd = [[YMARewardedAd alloc] initWithAdUnitID:<ваш уникальный AdUnitID>];  
self.rewardedAd.delegate = self;
```

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

При этом self должен удовлетворять протоколу [YMARewardedAdDelegate](#).

4. Загрузите рекламу.

Swift

```
rewardedAd.load()
```

Objective-C

```
[self.rewardedAd load];
```

Опционально, с помощью класса [YMAAdRequest](#), можно передать данные для таргетирования. Пример использования класса см. в разделе [Полноэкранная реклама](#).

Производить предзагрузку рекламы необходимо в той же ориентации, в которой она будет показана (иначе реклама не будет показана из-за несоответствия размеров экрана и баннера).

5. Начните отображение объявления, используя данный метод:

Swift

```
func rewardedAdDidLoad(_ rewardedAd: YMARewardedAd) {  
    rewardedAd.present(from: self)
```

}

Objective-C

```
- (void)rewardedAdDidLoad:(YMARewardedAd *)rewardedAd
{
    [rewardedAd presentFromViewController:self];
}
```

- Если вы используете механизм награждения на стороне клиента («client-side reward»), реализуйте метод делегата `-rewardedAd:didReward:`. Он вызывается, когда показ засчитан и пользователь может быть награжден за просмотр рекламы. Используйте этот момент, чтобы выдать награду пользователю приложения.
- Опционально можно включить логирование с помощью метода `+enableLogging`. Если показ рекламы не был засчитан, в консоли появится сообщение.

Чтобы определить причины, по которым реклама работает некорректно, используйте методы

Swift

```
func rewardedAdDidFail(toLoad rewardedAd: YMARewardedAd, error: Error)
func rewardedAdDidFail(toPresent rewardedAd: YMARewardedAd, error: Error)
```

Objective-C

```
- (void)rewardedAdDidFailToLoad:(YMARewardedAd *)rewardedAd error:(NSError *)error;
- (void)rewardedAdDidFailToPresent:(YMARewardedAd *)rewardedAd error:(NSError *)error;
```

Описание ошибок доступно в разделе [YMAAdErrorCode](#).

Чтобы посмотреть, как реклама будет отображаться в приложении, используйте демонстрационный AdUnitID:

```
let yandexAdUnitID = "demo-rewarded-yandex"
let adMobAdUnitID = "demo-rewarded-admob"
let appLovinAdUnitID = "demo-rewarded-applovin"
let ironSourceAdUnitID = "demo-rewarded-ironsource"
let mintegralAdUnitID = "demo-rewarded-mintegral"
let myTargetAdUnitID = "demo-rewarded-mytarget"
let startAppAdUnitID = "demo-rewarded-startapp"
let unityAdsAdUnitID = "demo-rewarded-unityads"
```

Понятия, связанные с данным

[Классы и протоколы для работы с видеорекламой с вознаграждением](#)

Узнайте больше

[Пример рекламы](#)

Нативная реклама

Процесс подключения

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Yandex Mobile Ads SDK предоставляет возможность показывать рекламу с использованием собственных визуальных компонентов.

Нативная реклама может изменяться в зависимости от функций и дизайна приложения, в котором она отображается. Оформление нативной рекламы соответствует среде, в которую она встраивается. Такая реклама выглядит органично и дополняет приложение полезной информацией.

SDK также предоставляет набор готовых настраиваемых визуальных компонентов (шаблонов), которые позволяют использовать все преимущества отрисовки нативными средствами платформы и не требуют создания собственного дизайна.

Yandex Mobile Ads SDK поддерживает несколько типов рекламы: App Install, Content, Image.

Чтобы подключить рекламу:

1. Ознакомьтесь с [требованиями к размещению рекламы и рекламных компонентов](#).
2. [Загрузите рекламу](#).
3. [Настройте внешнее оформление рекламы](#).

Понятия, связанные с данным

[Классы и протоколы для работы с нативной рекламой](#)

[Отслеживание работы рекламы](#)

Узнайте больше

[YMANativeErrorCode](#)

[Пример рекламы](#)

Загрузка рекламы



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Загрузка рекламных объявлений

1. Создайте экземпляр класса [YMANativeAdLoader](#) для получения нативных объявлений.
2. Создайте конфигурацию запроса `nativeAdRequestConfiguration` с помощью класса [YMANativeAdRequestConfiguration](#). В качестве параметров запроса можно передать идентификатор рекламного блока, способ загрузки изображений, возраст, гендерные признаки и другие данные, способные улучшить качество подбора рекламы.
3. Установите делегат для получения рекламы, который реализует протокол [YMANativeAdLoaderDelegate](#).
4. Чтобы отслеживать процесс загрузки рекламы, реализуйте методы протокола [YMANativeAdLoaderDelegate](#): `-nativeAdLoader:didFailLoadingWithError:`, `-nativeAdLoader:didLoadAd:`.
5. Отправьте загрузчику сообщение `loadAdWithRequestConfiguration:`, чтобы загрузить рекламу.

Swift

```
adLoader.loadAd(with: requestConfiguration)
```

Objective-C

```
[self.adLoader loadAdWithRequestConfiguration:requestConfiguration];
```

6. Если реклама загрузилась, будет вызван метод:

Swift

```
func nativeAdLoader(_ loader: YMANativeAdLoader, didLoad ad: YMANativeAd)
```

Objective-C

```
- (void)nativeAdLoader:(YMANativeAdLoader *)loader didLoadAd:(id<YMANativeAd>)ad
```

7. Если реклама не загрузилась, будет вызван следующий метод:

Swift

```
func nativeAdLoader(_ loader: YMANativeAdLoader, didFailLoadingWithError error: Error)
```

Objective-C

```
- (void)nativeAdLoader:(YMANativeAdLoader *)loader didFailLoadingWithError:(NSError *)error
```

Подробнее о возможных ошибках см. раздел [YMANativeErrorCode](#).

Примеры с демонстрационным adUnitID

Чтобы посмотреть, как реклама будет отображаться в приложении, используйте демонстрационный AdUnitID:

```
let yandexAdUnitID = "demo-native-content-yandex"
let adMobAdUnitID = "demo-native-admob"
let myTargetAdUnitID = "demo-native-mytarget"
```

Загрузка нескольких рекламных объявлений

Yandex Mobile Ads SDK предоставляет возможность загрузки нескольких рекламных объявлений одним запросом (до девяти объявлений).

1. Создайте экземпляр класса [YMANativeBulkAdLoader](#) для получения нативных объявлений.
2. Создайте конфигурацию запроса [nativeAdRequestConfiguration](#) с помощью класса [YMANativeAdRequestConfiguration](#). В качестве параметров запроса можно передать идентификатор рекламного блока, способ загрузки изображений, возраст, гендерные признаки и другие данные, способные улучшить качество подбора рекламы.
3. Установите делегат для получения рекламы, который реализует протокол [YMANativeBulkAdLoaderDelegate](#).
4. Чтобы отслеживать процесс загрузки рекламы, реализуйте методы протокола [YMANativeBulkAdLoaderDelegate](#): `-nativeBulkAdLoader:didLoadAds;`, `-nativeBulkAdLoader:didFailLoadingWithError;`.
5. Отправьте загрузчику конфигурацию запроса и количество запрашиваемых объявлений (параметр `adsCount`).

Swift

```
adLoader.loadAds(with: requestConfiguration, adsCount: adsCount)
```

Objective-C

```
[self.adLoader loadAdsWithRequestConfiguration:requestConfiguration adsCount:adsCount];
```

Swift

```
// Создание конфигурации запроса
let requestConfiguration = YMANutableNativeAdRequestConfiguration(adUnitID: AdUnitID)

// Создание загрузчика
adLoader = YMANativeBulkAdLoader()
adLoader.delegate = self

// Передача загрузчику конфигурации запроса и количество запрашиваемых рекламных объявлений
adLoader.loadAds(with: requestConfiguration, adsCount: adsCount)

// Реализация методов делегата

func nativeBulkAdLoader(_ nativeBulkAdLoader: YMANativeBulkAdLoader, didLoad ads: [YMANativeAd]) {
    ...
    // Отдельная работа с каждым объектом id<YMANativeAd>. Подробности в разделе
    "Настройка внешнего оформления рекламы".
}
```

Objective-C

```
// Создание конфигурации запроса
YMANativeAdRequestConfiguration *requestConfiguration =
    [[YMANativeAdRequestConfiguration alloc] initWithAdUnitID:AdUnitID];

// Создание загрузчика
self.adLoader = [[YMANativeBulkAdLoader alloc] init];
self.adLoader.delegate = self;

// Передача загрузчику конфигурации запроса и количество запрашиваемых рекламных объявлений
[self.adLoader loadAdsWithRequestConfiguration:requestConfiguration adsCount:adsCount];

// Реализация методов делегата

- (void)nativeBulkAdLoader:(YMANativeBulkAdLoader *)nativeBulkAdLoader didLoadAds:(NSArray<id<YMANativeAd>> *)ads
{
    ...
    // Отдельная работа с каждым объектом id<YMANativeAd>. Подробности в разделе
    "Настройка внешнего оформления рекламы".
}
```

```
}

```

Примечание:

Yandex Mobile Ads SDK не гарантирует, что будет загружено запрошенное количество объявлений. Полученный массив будет содержать от 0 до `adsCount` объектов `NativeAd`. Все полученные объекты рекламы можно показывать отдельно друг от друга, используя описанные выше способы внешнего оформления нативных объявлений.

Способы загрузки изображений**Автоматическая загрузка**

Если приложение одновременно хранит ссылки на одно или небольшое количество объявлений, рекомендуется использовать автоматическую загрузку.

Передайте `YES` в качестве значения свойства `shouldLoadImagesAutomatically` при создании конфигурации:

Swift

```
let requestConfiguration = YMAMutableNativeAdRequestConfiguration(adUnitID: AdUnitID)
requestConfiguration.shouldLoadImagesAutomatically = true

```

Objective-C

```
YMAMutableNativeAdRequestConfiguration *requestConfiguration =
[[YMAMutableNativeAdRequestConfiguration alloc] initWithAdUnitID:AdUnitID];
requestConfiguration.shouldLoadImagesAutomatically = YES;

```

В полученном объявлении будут присутствовать готовые изображения. Они хранятся в памяти приложения до уничтожения объявления.

Ручная загрузка

Если приложение одновременно хранит ссылки на большое количество объявлений, рекомендуется использовать ручную загрузку изображений.

Передайте `NO` в качестве значения свойства `shouldLoadImagesAutomatically` при создании конфигурации:

Swift

```
let requestConfiguration = YMAMutableNativeAdRequestConfiguration(adUnitID: AdUnitID)
requestConfiguration.shouldLoadImagesAutomatically = false

```

Objective-C

```
YMAMutableNativeAdRequestConfiguration *requestConfiguration =
[[YMAMutableNativeAdRequestConfiguration alloc] initWithAdUnitID:AdUnitID];
requestConfiguration.shouldLoadImagesAutomatically = NO;

```

В полученном объявлении будут присутствовать только размеры изображений. Чтобы загрузить сами изображения, вызовите метод `loadImages` у полученного объявления.

**Внимание:**

Все изображения кэшируются, но могут быть удалены в любой момент, поэтому необходимо вызывать метод `loadImages` перед каждым показом объявления.

Swift

```
func showAd() {
    // Show ad: custom native view or template
    view.addSubview(adView)
    adView.ad?.loadImages()
}

```

Objective-C

```
- (void)showAd
{
    // Show ad: custom native view or template
    [self.view addSubview:self.adView];
    [self.adView.ad loadImages];
}

```


Уведомления о процессе загрузки изображений

Ограничение: Получение уведомлений работает только при использовании ручной загрузки изображений.

Включить уведомления

Чтобы включить уведомления о завершении загрузки изображений, вызовите метод [-addImageLoadingObserver:](#)

Swift

```
ad?.add(self)

...
func nativeAdDidFinishLoadingImages(_ ad: YMANativeAd) {
    print("Finished loading images")
}
```

Objective-C

```
[ad addImageLoadingObserver:self];
...
-(void)nativeAdDidFinishLoadingImages:(id<YMANativeAd>)ad
{
    NSLog(@"Finished loading images");
}
```

Отключить уведомления

Чтобы отключить уведомления о завершении загрузки изображений, вызовите метод [-removeImageLoadingObserver:](#)

Swift

```
ad?.remove(self)
```

Objective-C

```
[ad removeImageLoadingObserver:self];
```

Настройка внешнего оформления рекламы



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Существует два способа настройки внешнего оформления рекламы:

1. Оформление с помощью шаблона.

Использование стандартного шаблона оформления — это самый простой способ работы с нативной рекламой. В шаблоне уже добавлен набор обязательных рекламных компонентов и настроено их расположение относительно друг друга. Шаблон работает с любым поддерживаемым типом рекламы.

2. Оформление без использования шаблона.

Оформление без шаблона используется в тех случаях, когда для настройки шаблона недостаточно функций, чтобы реклама выглядела органично и соответствовала дизайну приложения.

Оформление с помощью шаблона



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Для настройки внешнего оформления можно использовать стандартный шаблон оформления или создать свое оформление на основе стандартного шаблона.

Использование стандартного шаблона оформления

Примечание:

После настройки оформления задайте [местоположение и размеры рекламы](#) относительно экрана устройства.

1. Создайте объект — экземпляр класса `YMANativeBannerView` и установите для него загруженную рекламу:

Swift

```
let bannerView = YMANativeBannerView()
bannerView.ad = ad
view.addSubview(bannerView)
```

Objective-C

```
YMANativeBannerView *bannerView = [[YMANativeBannerView alloc] init];
bannerView.ad = ad;
[self.view addSubview:bannerView];
```

2. Чтобы получать уведомления о взаимодействии пользователя с рекламой (открытие рекламы, выход из приложения), установите для нее делегат `YMANativeAdDelegate`, реализующий методы:

- `-nativeAd:didDismissScreen;`
- `-nativeAd:willPresentScreen;`
- `-nativeAdWillLeaveApplication;`
- `-viewControllerForPresentingModalView.`

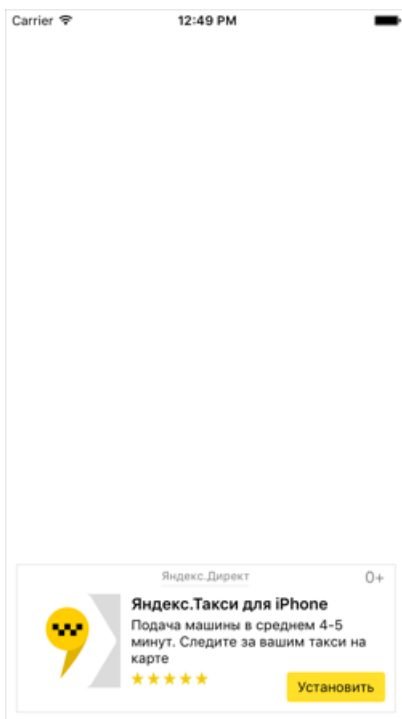
Swift

```
ad.delegate = self
```

Objective-C

```
ad.delegate = self;
```

3. Пример использования стандартного шаблона оформления:



Примечание: Если вашему приложению не подходит стандартное оформление, его можно изменить. Подробнее смотрите в разделе [Создание своего оформления на основе шаблона](#).

Создание своего оформления на основе шаблона

Примечание:

После настройки оформления задайте [местоположение и размеры рекламы](#) относительно экрана устройства.

1. Создайте объект — экземпляр класса [YMANativeBannerView](#) и установите для него загруженную рекламу:

Swift

```
let bannerView = YMANativeBannerView()
bannerView.ad = ad
view.addSubview(bannerView)
```

Objective-C

```
YMANativeBannerView *bannerView = [[YMANativeBannerView alloc] init];
bannerView.ad = ad;
[self.view addSubview:bannerView];
```

2. Чтобы получать уведомления о взаимодействии пользователя с рекламой (открытие рекламы, выход из приложения), установите для нее делегат [YMANativeAdDelegate](#), реализующий методы:

- [-nativeAd:didDismissScreen::](#);
- [-nativeAd:willPresentScreen::](#);
- [-nativeAdWillLeaveApplication::](#);
- [-viewControllerForPresentingModalView.](#)

Swift

```
ad.delegate = self
```

Objective-C

```
ad.delegate = self;
```

3. Запросите настройки стандартного шаблона оформления:

Swift

```
let appearance = YMAMutableNativeTemplateAppearance.default()
```

Objective-C

```
YMAMutableNativeTemplateAppearance *appearance = [[YMANativeTemplateAppearance defaultAppearance]
mutableCopy];
```

4. Задайте предпочитаемые настройки.
5. Чтобы применить настройки к шаблону, вызовите метод [-applyAppearance::](#):

Swift

```
bannerView.apply(appearance)
```

Objective-C

```
[bannerView applyAppearance:appearance];
```

Пример настроек оформления**Swift**

```
// Определяем собственные цветовые решения для работы.
let orangeColor = UIColor(red: 1, green: 176.0/255, blue: 32.0/255, alpha: 1)
let blueColor = UIColor(red: 0, green: 170.0/255, blue: 1, alpha: 1)

// Создаем копию с настройками стандартного шаблона оформления.
let appearance = YMAMutableNativeTemplateAppearance.default()

// Начинаем изменять стандартные настройки шаблона.

// Задаем цвет для рамки рекламного объявления.
appearance.borderColor = orangeColor

// Создаем копию с настройками рейтинга.
let ratingAppearance = appearance.ratingAppearance?.mutableCopy() as? YMAMutableRatingAppearance

// Задаем цвет для закрашенных звезд в рейтинге.
ratingAppearance?.filledStarColor = orangeColor
appearance.ratingAppearance = ratingAppearance

// Задаем цвет и размер шрифта для надписи на кнопке с действием.
```

```

let callToActionTextAppearance = YMALabelAppearance(font: .systemFont(ofSize: 14), textColor: blueColor)

// Задаем цвет кнопки для обычного и нажатого состояния, цвет и толщину обводки кнопки.
let callToActionAppearance = YMAButtonAppearance(
    textAppearance: callToActionTextAppearance,
    normalColor: .clear,
    highlightedColor: .gray,
    borderColor: blueColor,
    borderWidth: 1
)
appearance.callToActionAppearance = callToActionAppearance

// Задаем размер и цвет шрифта для надписи с возрастным ограничением.
appearance.ageAppearance = YMALabelAppearance(font: .systemFont(ofSize: 12), textColor: .gray)

// Задаем размер и цвет шрифта для заголовка рекламного объявления.
appearance.titleAppearance = YMALabelAppearance(font: .systemFont(ofSize: 14), textColor: .black)

// Задаем размер и цвет шрифта для основного рекламного текста.
appearance.bodyAppearance = YMALabelAppearance(font: .systemFont(ofSize: 12), textColor: .gray)

// Задаем ширину изображения и правило формирования размера.
let imageConstraint = YMASizeConstraint(type: .fixed, value: 60)

// Применяем настройки к изображению.
appearance.imageAppearance = YMAImageAppearance(widthConstraint: imageConstraint)

```

Objective-C

```

// Определяем собственные цветовые решения для работы.
UIColor *orangeColor =
    [UIColor colorWithRed:255.f / 255.f green:176.f / 255.f blue:32.f / 255.f alpha:1.f];
UIColor *blueColor =
    [UIColor colorWithRed:0.f / 255.f green:170.f / 255.f blue:255.f / 255.f alpha:1.f];

// Создаем копию с настройками стандартного шаблона оформления.
YMAMutableNativeTemplateAppearance *appearance = [[YMANativeTemplateAppearance defaultAppearance] mutableCopy];

// Начинаем изменять стандартные настройки шаблона.

// Задаем цвет для рамки рекламного объявления.
appearance.borderColor = orangeColor;

// Создаем копию с настройками рейтинга.
YMAMutableRatingAppearance *ratingAppearance = [appearance.ratingAppearance mutableCopy];

// Задаем цвет для закрашенных звезд в рейтинге.
ratingAppearance.filledStarColor = orangeColor;
appearance.ratingAppearance = ratingAppearance;

// Задаем цвет и размер шрифта для надписи на кнопке с действием.
YMALabelAppearance *callToActionTextAppearance =
    [YMALabelAppearance appearanceWithFont:[UIFont systemFontOfSize:14.f]
    textColor:blueColor];

// Задаем цвет кнопки для обычного и нажатого состояния, цвет и толщину обводки кнопки.
YMAButtonAppearance *callToActionAppearance =
    [YMAButtonAppearance appearanceWithTextAppearance:callToActionTextAppearance
    normalColor:[UIColor clearColor]
    highlightedColor:[UIColor grayColor]
    borderColor:blueColor
    borderWidth:1.f];
appearance.callToActionAppearance = callToActionAppearance;

// задаем размер и цвет шрифта для надписи с возрастным ограничением.
appearance.ageAppearance =
    [YMALabelAppearance appearanceWithFont:[UIFont systemFontOfSize:12.f]
    textColor:[UIColor grayColor]];

// Задаем размер и цвет шрифта для заголовка рекламного объявления.
appearance.titleAppearance =
    [YMALabelAppearance appearanceWithFont:[UIFont systemFontOfSize:14.f]
    textColor:[UIColor blackColor]];

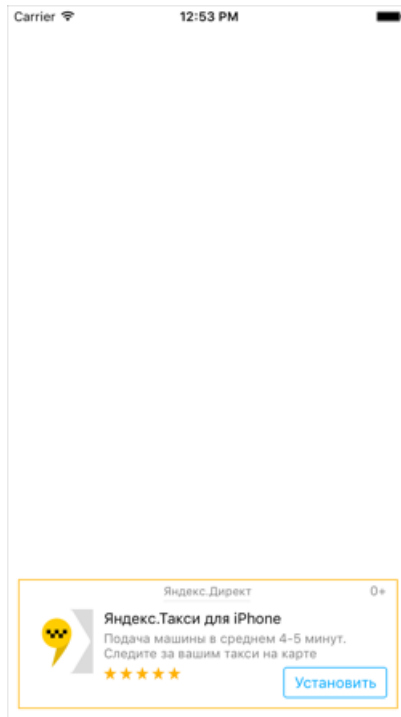
// Задаем размер и цвет шрифта для основного рекламного текста.
appearance.bodyAppearance =
    [YMALabelAppearance appearanceWithFont:[UIFont systemFontOfSize:12.f]
    textColor:[UIColor grayColor]];

// Задаем ширину изображения и правило формирования размера.
YMASizeConstraint *imageConstraint =
    [YMASizeConstraint constraintWithType:YMASizeConstraintTypeFixed value:60.f];

// Применяем настройки к изображению.
appearance.imageAppearance =
    [YMAImageAppearance appearanceWithWidthConstraint:imageConstraint];

```

Получаем собственный дизайн на основе шаблона:



Настройка местоположения и размера рекламы

Ограничение: Требования к размерам `mediaView` при отображении видеорекламы

Минимальный размер экземпляра класса `YMANativeMediaView`, в котором поддерживается воспроизведение видео: 300x160 или 160x300.

Для поддержки воспроизведения видео в шаблонах нативной рекламы рекомендуется выставить ширину для `YMANativeBannerView` не менее 300. Корректная высота для `mediaView` будет вычислена автоматически, с учетом соотношения ширины и высоты.

Управлять размерами и местоположением можно одним из двух способов:

1. с помощью системного механизма `AutoLayout`;

Примечание:

При работе с `AutoLayout` задайте `constraint` для ширины `UIView`. Высота будет определена автоматически на основании указанной ширины.

2. вручную, задавая все размеры.

Настройка размеров вручную

Примечание:

Не рекомендуется задавать ширину объекта более 420 логических пикселей.

Задайте для объекта `YMANativeBannerView` ширину и высоту. Высота определяется с помощью метода `+heightWithAd:width:appearance:`, в который необходимо передать рекламное объявление, ширину и объект `YMANativeTemplateAppearance` (используется для настроек внешнего вида рекламного объявления).

Swift

```
// Задаем отступы слева и справа относительно экрана.
let inset: CGFloat = 50

// Задаем ширину.
let width = view.frame.width - 2 * inset

// Задаем высоту.
let height = YMANativeBannerView.height(with: ad, width: width, appearance: nil)

// Задаем позицию по вертикали.
let y = view.frame.maxY - height - inset

// Задаем координаты и размеры для frame.
```

```
bannerView.frame = CGRect(x: inset, y: y, width: width, height: height)
```

Objective-C

```
// Задаем отступы слева и справа относительно экрана.
CGFloat inset = 50.f;

// Задаем ширину.
CGFloat width = CGRectGetWidth(self.view.frame) - 2 * inset;

// Задаем высоту.
CGFloat height = [YMANativeBannerView heightForAd:ad width:width appearance:nil];

// Задаем позицию по вертикали.
CGFloat y = CGRectGetMaxY(self.view.frame) - height - inset;

// Задаем координаты и размеры для frame.
bannerView.frame = CGRectMake(inset, y, width, height);
```

Оформление без использования шаблона



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Для отображения нативной рекламы объекты `YMANativeAd` должны быть привязаны к определенной `View`. Эта `View` должна определяться как корневой элемент для набора `subview`, к элементам которого будут привязываться данные, содержащиеся в нативной рекламе.

Ограничение: Требования к размерам `mediaView` при отображении видеорекламы

Минимальный размер экземпляра класса `YMANativeMediaView`, в котором поддерживается воспроизведение видео: 300x160 или 160x300.

Для поддержки воспроизведения видео в нативной рекламе рекомендуется выставить ширину для `mediaView` не менее 300. Чтобы вычислить соответствующее значение высоты `mediaView`, используйте значение свойства `aspectRatio`.

Оформление нативной рекламы

1. Ознакомьтесь со [списком](#) обязательных и опциональных `subview` для каждого типа `View`.
2. Установите значения всех обязательных `subview`. Это можно сделать с помощью `nib` или непосредственно в коде. Для этого создайте объект класса `YMANativeAdView` и определите значения `subview` в `initWithFrame`:

Пример регистрации набора `subview`:

Swift

```
override init(frame: CGRect) {
    super.init(frame: frame)
    let titleLabel = createLabel()
    let bodyLabel = createLabel()
    let ageLabel = createLabel()
    let warningLabel = createLabel()
    let sponsoredByLabel = createLabel()
    let priceLabel = createLabel()
    let starRatingView = createStarRatingView()
    let button = createButton()
    let iconImageView = createIconAssetImageView()
    let mediaView = createMediaAssetView()
    addSubview(titleLabel)
    addSubview(bodyLabel)
    addSubview(ageLabel)
    addSubview(warningLabel)
    addSubview(sponsoredByLabel)
    addSubview(priceLabel)
    addSubview(starRatingView)
    addSubview(button)
    addSubview(iconImageView)
    addSubview(mediaView)
    self.titleLabel = titleLabel
    self.bodyLabel = bodyLabel
    self.ageLabel = ageLabel
    self.warningLabel = warningLabel
    self.sponsoredLabel = sponsoredByLabel
    self.callToActionButton = callToActionButton
    self.priceLabel = priceLabel
    self.ratingView = ratingView
    self.iconImageView = iconImageView
    self.mediaView = mediaView
}
```

```
}
}
```

Objective-C

```
- (instancetype)initWithFrame:(CGRect)frame
{
    self = [super initWithFrame:frame];
    if (self != nil) {
        UILabel *titleLabel = [self label];
        UILabel *bodyLabel = [self label];
        UILabel *ageLabel = [self secondaryLabel];
        UILabel *warningLabel = [self label];
        UILabel *sponsoredByLabel = [self label];
        UILabel *priceLabel = [self label];
        StarRatingView *ratingView = [self starRatingView];
        UIButton *callToActionButton = [self button];
        UIImageView *iconImageView = [self iconAssetImageView];
        YMANativeMediaView *mediaView = [self mediaAssetView];
        [self addSubview:titleLabel];
        [self addSubview:bodyLabel];
        [self addSubview:ageLabel];
        [self addSubview:warningLabel];
        [self addSubview:sponsoredByLabel];
        [self addSubview:callToActionButton];
        [self addSubview:priceLabel];
        [self addSubview:ratingView];
        [self addSubview:iconImageView];
        [self addSubview:mediaView];
        self.titleLabel = titleLabel;
        self.bodyLabel = bodyLabel;
        self.ageLabel = ageLabel;
        self.warningLabel = warningLabel;
        self.sponsoredLabel = sponsoredByLabel;
        self.callToActionButton = callToActionButton;
        self.priceLabel = priceLabel;
        self.ratingView = ratingView;
        self.iconImageView = iconImageView;
        self.mediaView = mediaView;
    }
    return self;
}
```

- Чтобы получать уведомления о взаимодействии пользователя с рекламой (открытие и закрытие рекламы, выход из приложения), установите для нее делегат [YMANativeAdDelegate](#), реализующий методы:
 - closeNativeAd;
 - nativeAd:didDismissScreen;
 - nativeAd:willPresentScreen;
 - nativeAdWillLeaveApplication;
 - viewControllerForPresentingModalView.
- Запросите значения компонентов нативной рекламы с помощью метода [-adAssets](#). Это поможет заранее рассчитать расположение и размеры этих компонентов.

Swift

```
let assets = ad.adAssets()
```

Objective-C

```
YMANativeAdAssets *assets = [ad adAssets];
```

Пример получения размеров изображения, соотношения сторон медиа и текста заголовка объявления**Swift**

```
let image = assets.image
let title = assets.title
let media = assets.media
print("Image size: \(image?.size ?? .zero)")
print("Title: \(title ?? "")")
print(String(format: "Media aspect ratio: %.2f", media?.aspectRatio ?? 0))
```

Objective-C

```
YMANativeAdImage *image = assets.image;
NSString *title = assets.title;
YMANativeAdMedia *media = assets.media;
NSLog(@"Image size: %@", NSStringFromCGSize(image.size));
NSLog(@"Title: %@", title);
NSLog(@"Media aspect ratio: %.2f", media.aspectRatio);
```

5. Вызовите метод `-bindWithAdView:error:`, чтобы связать контент с объектом нативной рекламы.

Swift

```
// ...
adView = YMANativeAdView(frame: frame)
//configure content ad view
// ...
func nativeAdLoader(_ loader: YMANativeAdLoader, didLoad ad: YMANativeAd) {
    ad.delegate = self
    do {
        try ad.bind(with: contentAdView)
    } catch {
        print("Error: \(error)")
    }
}
```

Objective-C

```
// ...
self.adView = [[YMANativeAdView alloc] initWithFrame:frame];
//configure content ad view
// ...
- (void)nativeAdLoader:(YMANativeAdLoader *)loader didLoadAd:(id<YMANativeAd>)ad
{
    ad.delegate = self;
    NSError *error = nil;
    BOOL result = [ad bindWithAdView:self.contentAdView error:error];
    if (error != nil) {
        NSLog(@"Error: %@", error);
    }
}
```

Примечание:

Если для обязательного элемента нативной рекламы соответствующее свойство `YMANativeAdView` принимает значение `nil`, привязка не осуществится — реклама не будет показана. Детали в этом случае можно получить из `error`.

Отладка



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Логирование

Опционально можно включить логирование с помощью метода `+enableLogging`. Если показ рекламы не был засчитан, в консоли появится сообщение.

Индикатор некорректной интеграции для нативной рекламы

Если при интеграции нативной рекламы допущена ошибка, то в режиме симулятора, поверх рекламного объявления, появится индикатор. По нажатию на индикатор вы увидите сообщение с отладочной информацией, по которой можно понять причину ошибки. Повторное нажатие на индикатор скрывает сообщение.



Внимание:

По умолчанию индикатор отображается только в режиме симулятора (тип устройства `YMADeviceTypeSimulator`). Типы устройства можно посмотреть в [YMADeviceType](#).

Чтобы включить индикатор также и для реальных устройств, передайте значение `YMADeviceTypeHardware | YMADeviceTypeSimulator` в методе `enableVisibilityErrorIndicatorForDeviceType::`

Swift

```
YMAMobileAds.enableVisibilityErrorIndicator(for: [.hardware, .simulator])
```

Objective-C

```
[YMAMobileAds enableVisibilityErrorIndicatorForDeviceType:YMADeviceTypeHardware | YMADeviceTypeSimulator]
```

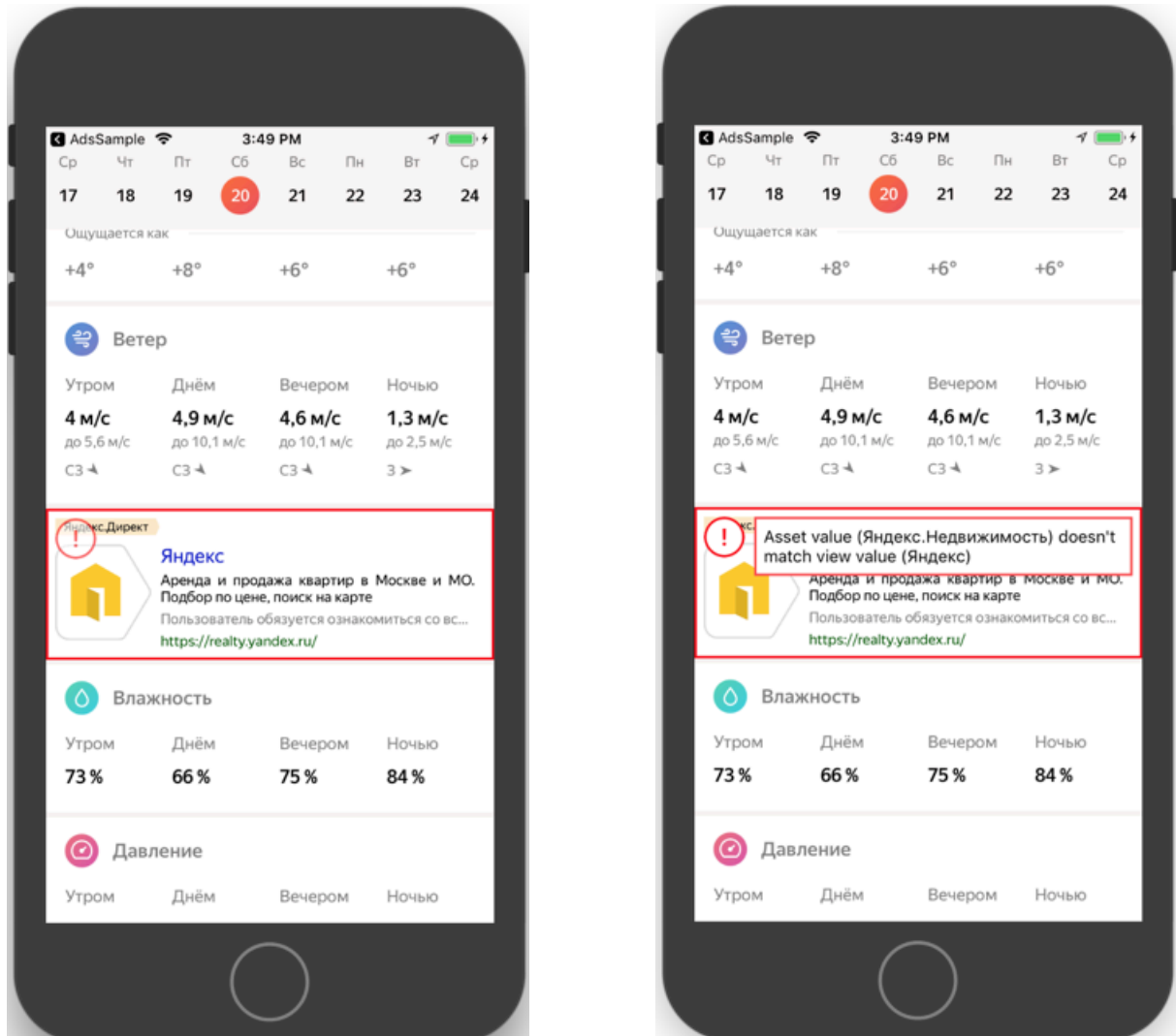

Чтобы выключить индикатор, передайте значение `YMADeviceTypeNone` в методе `enableVisibilityErrorIndicatorForDeviceType::`

Swift

```
YMAMobileAds.enableVisibilityErrorIndicator(for: [])
```

Objective-C

```
[YMAMobileAds enableVisibilityErrorIndicatorForDeviceType:YMADeviceTypeNone]
```



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 | Версия адаптера | Версия SDK рекламной сети | Баннерная реклама (Banner Ad) | Полноэкранная реклама (Interstitial Ad) | Реклама с вознаграждением (Rewarded Ad) | Нативная реклама (Native Ad) |
|----------------------------|-----------------|---------------------------|-------------------------------|---|---|------------------------------|
| AdMob | 10.7.0.0 | 10.7.0 | ✓ | ✓ | ✓ | ✓ |
| myTarget | 5.17.5.1 | 5.17.5 | ✓ | ✓ | ✓ | ✓ |
| Start.io | 4.7.3.0 | 4.7.3 | ✓ | ✓ | ✓ | ✗ |
| UnityAds | 4.8.0.0 | 4.8.0 | ✓ | ✓ | ✓ | ✗ |
| AppLovin | 11.10.1.1 | 11.10.1 | ✗ | ✓ | ✓ | ✗ |
| IronSource | 7.3.0.3 | 7.3.0 | ✗ | ✓ | ✓ | ✗ |
| Mintegral | 7.3.9.0 | 7.3.9 | ✓ | ✓ | ✓ | ✗ |
| AdColony | 4.9.0.4 | 4.9.0 | ✓ | ✓ | ✓ | ✗ |
| Chartboost | 9.3.1.0 | 9.3.1 | ✓ | ✓ | ✓ | ✗ |

Подключение AdMob

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

GoogleMobileAdsSDK

Минимальная поддерживаемая версия: 10.7.0.

Максимальная поддерживаемая версия: 11.0.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Баннерная реклама](#) (Banner)
- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)
- [Нативная реклама](#) (Native)

Подключение

Для работы с [AdMob](#) необходимо подключить библиотеку `AdMobYandexMobileAdsAdapters`.

Библиотека `AdMobYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в `Podfile` проекта зависимость:

```
pod 'AdMobYandexMobileAdsAdapters', '10.7.0.0'
```

3. В своем приложении, в файле `Info.plist`, добавьте ключ `GADApplicationIdentifier` с вашим `AdMob ID` (подробнее о том, где [найти AdMob ID](#)).

```
<key>GADApplicationIdentifier</key>  
<string>ca-app-pub-xxxxxxxxxxxxxxxx~yyyyyyyyyy</string>
```

Примеры интеграции

- [Пример рекламы](#)

Подключение myTarget



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

myTargetSDK

Минимальная поддерживаемая версия: 5.17.5.

Максимальная поддерживаемая версия: 5.18.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Баннерная реклама](#) (Banner)
- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)
- [Нативная реклама](#) (Native)

Подключение

Для работы с [myTarget](#) необходимо подключить библиотеку `MyTargetYandexMobileAdsAdapters`.

Библиотека `MyTargetYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.

2. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимость:

```
pod 'MyTargetYandexMobileAdsAdapters', '5.17.5.1'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение Start.io



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать Info.plist для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.2.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

Start.io SDK

Минимальная поддерживаемая версия: 4.7.3.

Максимальная поддерживаемая версия: 4.8.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

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

Подключение

Для работы с [Start.io](#) необходимо подключить библиотеку StartAppYandexMobileAdsAdapters.

Библиотека StartAppYandexMobileAdsAdapters адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения.

Примечание:

Перед началом работы медиационная сеть Start.io отображает всплывающее окно с запросом согласия пользователя на персонализацию рекламы (подробнее в [документации сети Start.io](#)). Если вы хотите скрывать всплывающее окно, передайте согласие пользователя на персонализацию рекламы с помощью метода + `setUserConsent`:

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимость:

```
pod 'StartAppYandexMobileAdsAdapters', '4.7.3.0'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение UnityAds

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

UnityAdsSDK

Минимальная поддерживаемая версия: 4.8.0.

Максимальная поддерживаемая версия: 4.9.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

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

Подключение

Для работы с [UnityAds](#) необходимо подключить библиотеку `UnityYandexMobileAdsAdapters`.

Библиотека `UnityYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в `Podfile` проекта зависимость:

```
pod 'UnityAdsYandexMobileAdsAdapters', '4.8.0.0'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение AppLovin

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

AppLovinSDK

Минимальная поддерживаемая версия: 11.10.1.

Максимальная поддерживаемая версия: 11.11.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)

Подключение

Для работы с [AppLovin](#) необходимо подключить библиотеку AppLovinYandexMobileAdsAdapters.

Библиотека AppLovinYandexMobileAdsAdapters адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимость:

```
pod 'AppLovinYandexMobileAdsAdapters', '11.10.1.1'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение IronSource



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать Info.plist для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

IronSourceSDK

Минимальная поддерживаемая версия: 7.3.0.

Максимальная поддерживаемая версия: 7.4.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)

Подключение

Для работы с [IronSource](#) необходимо подключить библиотеку IronSourceYandexMobileAdsAdapters.

Библиотека `IronSourceYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в `Podfile` проекта зависимость:

```
pod 'IronSourceYandexMobileAdsAdapters', '7.3.0.3'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение Mintegral



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

MintegralSDK

Минимальная поддерживаемая версия: 7.3.9.

Максимальная поддерживаемая версия: 7.4.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

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

Подключение

Для работы с `Mintegral` необходимо подключить библиотеку `MintegralYandexMobileAdsAdapters`.

Библиотека `MintegralYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в `Podfile` проекта зависимость:

```
pod 'MintegralYandexMobileAdsAdapters', '7.3.9.0'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение AdColony

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

AdColonySDK

Минимальная поддерживаемая версия: 4.9.0.

Максимальная поддерживаемая версия: 4.10.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Баннерная реклама](#)
- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)

Подключение

Для работы с [AdColony](#) необходимо подключить библиотеку `AdColonyYandexMobileAdsAdapters`.

Библиотека `AdcolonyYandexMobileAdsAdapters` адаптирована для работы с системой управления зависимостями `CocoaPods` и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в `Podfile` проекта зависимость:

```
pod 'AdColonyYandexMobileAdsAdapters', '4.9.0.4'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение Chartboost

**Внимание:**

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Примечание:

Для трекинга установок приложений в мобильной медиации вам необходимо сконфигурировать `Info.plist` для адаптера. Следуйте [документации](#) рекламной сети.

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

ChartboostSDK

Минимальная поддерживаемая версия: 9.3.1.

Максимальная поддерживаемая версия: 9.4.0 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Баннерная реклама](#)
- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)

Подключение

Для работы с [Chartboost](#) необходимо подключить библиотеку ChartboostYandexMobileAdsAdapters.

Библиотека ChartboostYandexMobileAdsAdapters адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимость:

```
pod 'ChartboostYandexMobileAdsAdapters', '9.3.1.0'
```

Примеры интеграции

- [Пример рекламы](#)

Подключение VigoADS



Внимание:

Это архивная версия документации. Актуальная документация по всем платформам находится [здесь](#).

Поддерживаемые версии библиотек и платформ:

YandexMobileAdsSDK

Минимальная поддерживаемая версия: 5.9.0.

Максимальная поддерживаемая версия: 6.0.0 (не включая).

VigoADSSDK

Минимальная поддерживаемая версия: 2.3.2.

Максимальная поддерживаемая версия: 2.3.2 (не включая).

iOS

Минимальная поддерживаемая версия: 13.0.

Поддерживаемые форматы рекламы

- [Баннерная реклама](#)
- [Полноэкранная реклама](#) (Interstitial)
- [Реклама с вознаграждением](#) (Rewarded)

Подключение

Для работы с [VigoADS](#) необходимо подключить библиотеку VigoADSYandexMobileAdsAdapters.

Библиотека BigoADSYandexMobileAdsAdapters адаптирована для работы с системой управления зависимостями CocoaPods и поддерживает статический способ подключения.

1. [Настройте медиацию](#) в Партнерском интерфейсе и в кабинете рекламной сети.
2. Чтобы подключить библиотеку, добавьте в Podfile проекта зависимость:

```
pod 'BigoADSYandexMobileAdsAdapters', '2.3.2.0'
```

Примеры интеграции

- [Пример рекламы](#)