



ziginsider

23 дек 2017 в 14:59

Полное руководство по Splash Screen на Android



9 мин



80К

Разработка мобильных приложений*, Разработка под Android*, Kotlin*

Из песочницы

Перевод статьи Elvis Chidera "The (Complete) Android Splash Screen Guide". Но для начала немного истории о проблеме Splash Screen на Android.

Два вида Splash Screen

Google в гайдлайнах Material Design представляет две концепции Splash Screen (или Launch Screen):

1. Placeholder UI
2. Branded launch screen

Placeholder UI — концепция для приложений, которые загружаются довольно быстро и не требуют показа перед запуском логотипов, красивых анимаций и прочих украшений. Суть в том, что во время холодного старта основной фон приложения, строка состояния, фон для панели инструментов уже раскрашиваются в цвета приложения, но до полной загрузки без контента. Такой подход, по мнению дизайнеров Google, позволяет более плавно с точки зрения пользователя переходить от момента запуска приложения к работе с ним.

▶ [Пример Placeholder UI](#)

Branded launch screen — собственно и есть то, что большинство разработчиков (по крайней мере Android-разработчиков), именуют Splash Screen. Это некоторый логотип, изображение, реже анимация, которые пользователь на короткое время видит во время старта приложения.

▶ [Пример Branded launch screen](#)

Теперь, прежде чем перейдем к переводу статьи Элвиса, которая рассказывает о Branded launch screen, немного о грустном в лагере поклонников Placeholder UI.

Placeholder не работает даже у приложений от Google

Не смотря на собственные гайдлайны, Google не смог реализовать подход Placeholder UI в собственных приложениях. Я навскидку выбрал три популярных приложения от Google, где по логике должен быть Placeholder UI, но он не работает. Показан переход от холодного старта к рабочему состоянию приложения:

▶ [Gmail](#)

▶ [Google Keep](#)

▶ [Play Market](#)

Как видим, во время холодного старта загружается только фон приложения, панель инструментов и строка состояния — либо под цвет фона, либо случайного цвета (как в примере с Play Market).

Причины этому следующие:

1. Сегодня хорошим тоном считается использовать `Toolbar`, как часть макета панели инструментов. Это дает много плюшек: реакция на прокрутку, анимации и т.д. Но вместе с тем, необходимо использовать тему `NoActionBar`. Это влияет на то, что цвета главной темы приложения не подгружаются при холодном старте.
2. Проблема в библиотеке `AppCompat`. Даже если использовать обычный `ActionBar` для панели инструментов, мы будем наблюдать аналогичный эффект. Переход от `@style/Theme.AppCompat.Light.DarkActionBar` к `@android:style/Theme.Material.Light.DarkActionBar` лечит эту проблему, но, вероятно, приложения все еще рассчитаны на поддержку версий до Lollipop.

Впрочем, есть решение.

О подходе Placeholder UI все. Переходим к переводу статьи Элвиса.

Как работать с Branded launch screen

В прошлом в Android не рекомендовалось делать Splash Screen в приложениях. Не было смысла заставлять пользователя ждать n секунд, чтобы показать заставку. И несомненно, что никто не запускает приложение ради заставки (об этом ниже).

Когда в Material Design появился раздел под названием Launch Screen (Splash Screen), кто-то из команды Android опубликовал [пост](#) о том, как сделать Splash Screen правильно.


В этом посте я рассмотрю четыре основных способа реализации Splash Screen на Android:

1. Используя Launcher Theme (**Хорошо**)
2. Используя Launcher Theme с предопределенной для Splash Screen Activity (**Сойдет**)
3. Используя таймеры (Timers) (**Плохо**)
4. Используя умные таймеры (Smart Timers) (**Ужасно**)

Используя Launcher Theme

Когда ваше приложение запускается и оно еще не в памяти, может иметь место задержка между тем, когда пользователь нажал на запуск, и тем, когда у Activity вызвано `onCreate()`. Этот, так называемый "холодный старт" — лучшее время, чтобы показать ваш Splash Screen.

Во время "холодного старта" оконный менеджер пытается отрисовать placeholder UI, используя элементы из темы приложения (app theme), такие как `windowBackground`. И то, что показывает `windowBackground` по-умолчанию (обычно белый или черный фон), вы можете поменять на какой

 +12

 132



 15

Итак, вам необходимо создать кастомную тему, переопределив `android:windowBackground`, заменив использование стандартной темы на вашу перед вызовом `super.onCreate` в вашей Activity.

В этом примере, я предполагаю, что главная тема вашего приложения называется **AppTheme**, но если это не так, просто во всех местах замените **AppTheme** на имя главной темы вашего приложения.

Вы должны создать новую тему `AppTheme.Launcher`. Единственный элемент, который необходимо переопределить — это `windowBackground`. В файл `styles.xml` добавим:

► [КОД](#)

При этом мы наследуем все остальные атрибуты главной темы **AppTheme**, используя ее название, как префикс для названия нашей темы **Launcher**.

Определяем drawable `launch_screen`. Хотя вы могли бы использовать простую картинку, но она будет растянута на весь экран. Вместо этого используем XML-файл:

► КОД

Пропишите тему для Splash Screen в файле манифеста в вашей стартовой Activity:

```
<activity android:theme="@style/AppTheme.Launcher">
```

Теперь нужно вернуть главную тему в стартовую Activity (если, конечно, мы не хотим, чтобы Splash Screen радовал нас и во время работы приложения)

Самый простой способ сделать это — это вызвать `setTheme(R.style.AppTheme)` до `super.onCreate()` и `setContentView()`:

► КОД

Все. Вы можете узнать подробнее об этом подходе [здесь](#).

Плюсы:

1. Нет никакой специальной Activity для Splash Screen. Нет задержки времени как в том случае, если бы вы вызывали рабочую Activity из Activity для Splash Screen.
2. Нет искусственных задержек — Splash Screen показывается только тогда, когда приложение загружается.

Минусы:

Я встречал три довольно распространенные жалобы на этот подход:

1. Splash Screen показывается снова если Activity была убита системой и снова восстановлена. В большинстве случаев, этот совсем не проблема, но при желании, если использовать **второй способ**, этого можно избежать.
2. Некоторым разработчикам необходимо, чтобы после запуска Splash Screen, пользователь попадал на разные Activity, в зависимости от каких-либо параметров. Опять же, для таких задач можно использовать **второй способ**, но иногда реализация такой промежуточной Activity довольно неряшлива.
3. Невозможно загрузить тяжелые данные/компоненты, когда показывается Splash Screen. Хотя это плохая идея: загружать тяжелые данные или компоненты, пока стартует приложение (есть некоторые исключения, как, например, инициализация некоторых библиотек). Можно попробовать один из следующих подходов.

- Постарайтесь использовать метод ленивой загрузки, для своих компонентов/модулей/библиотек. За исключением компонентов, которые критически необходимы для работы приложения, старайтесь ничего не загружать во время запуска приложения, а загружать, когда вам понадобится компонент, или грузите в фоновом потоке сразу после старта приложения. Сохраняйте `onCreate()` вашего приложения легким насколько это возможно.
- Используйте кэширование. За исключением той информации, которая быстро меняется, остальное лучше кэшировать. Когда пользователь снова обращается к вашему приложению, вы можете показать закэшированный контент, пока более свежий контент загружается.

Я думаю, что следует избегать таких вещей, как долгий Splash Screen, как ProgressDialog, которые заставляют пользователя просто смотреть на экран и не дают ему выполнить никакое действие.

Если ваше приложение подключается к сети, предположите, что все, что должно пойти не так, пойдет не так. Таким образом вы сможете создавать приложения для миллионов людей, которые все еще используют нестабильные соединения 2G и 3G

Splash Screen в отдельной Activity с использованием Launcher Theme

Этот способ базируется на **первом способе**. Он требует отдельной Activity для Splash Screen. Первые два шага пропускаем, они аналогичны **первому способу**.

Осталось создать Activity для Splash Screen и указать в манифесте для нее тему `AppTheme.Launcher`. Теперь отредактируем Activity так, чтобы она перенаправляла на другие страницы. Смотрим пример ниже:

► [КОД](#)

Плюсы:

1. Решает первые две проблемы у **первого способа**.

Минусы:

1. Я видел, как подобная маршрутизация легко становится уродливой.
2. Небольшая задержка между двумя Activity.
3. Опасность забыть и начать делать длительные операции здесь.

Используя таймеры

Это старый добрый подход. Надо просто создать отдельную для Splash Screen Activity, которая будет показываться x секунд. Затем открыть подходящую Activity. Используя такой подход, вы получаете больше гибкости, потому что можете добавить анимацию, кастомные view или любые другие элементы, которые вы можете поместить в макет Activity. Вот минимальная реализация такого подхода:

► [КОД](#)

Плюсы:

1. Появляется возможность показать вашу супер-анимацию, или любой другой кастомный дизайн, который вы хотите. Это имеет смысл для игровых приложений или приложений для детей.
2. Большая гибкость того, что вы можете показать на Splash Screen.

Минусы:

1. Двойной удар — ваша Activity не стартовая Activity не появляется немедленно, после того, как приложение запущено, особенно во время холодного старта. Пользователь ждет во время холодного старта, наблюдая только `windowBackground` и, затем, просматривая еще Splash Screen до того, как запустится рабочая Activity.
2. Ваша супер-анимация или дизайн обычно восхищает пользователя только первые пару раз. Потом, большинство пользователей находит его скучным, и они хотят получить только контент. Я думаю, что **четвертый способ** способен исправить это.
3. В большинстве случаев дополнительная задержка неоправданна.

Используя умные таймеры.

Этот подход базируется на **третьем способе**. Но вместо постоянной задержки, вы запускаете Splash Screen или нет, основываясь на том, первый это запуск или нет. Вот пример, который использует `SharedPreferences` :

► [КОД](#)

Плюсы:

1. Это возможное решение проблемы, когда пользователь устает наблюдать ваш Splash Screen в течении долгого времени.

Минусы:

1. Двойной удар — проблемы **третьего способа** все еще не здесь
2. В большинстве случаев дополнительная задержка неоправданна.
3. Я не использовал этот метод, но, думаю, может быть некоторая задержка при чтении из `SharedPreferences` .

Это все о Splash Screen. Если я что-то упустил, напишите в комментариях.

Следует заметить, что на Хабре уже была [статья](#) (перевод), где речь шла о Splash Screen. Однако затронутый там подход (соответствует второму способу в этой статье), как мы могли убедиться, не самый оптимальный для большинства случаев. Ну и последнее, в Android Oreo якобы [добавлено](#) Splash Screen API, что позволит разработчикам легко добавлять Splash Screen в свои приложения, но на данный момент в официальной документации по этому поводу никакой информации нет.

Теги: splash screen, launch screens, android, kotlin

Хабы: Разработка мобильных приложений, Разработка под Android, Kotlin

Редакторский дайджест

Присылаем лучшие статьи раз в месяц

Электронпочта



8

Карма

0

Рейтинг

Alex Kisel @ziginsider

Пользователь

1cloud
Удаленный
VDS/VPS-сервер
в аренду
за 2 минуты!
Подключить

Удаленный сервер в аренду. Бесплатное тестирование

Система виртуализации Enterprise-уровня. SLA 99,96%. Бесплатный мониторинг. ТП 24/7

Комментарии 15

Публикации

ЛУЧШИЕ ЗА СУТКИ ПОХОЖИЕ



Kekovsky

10 часов назад

Как я компьютер «Радио 86-РК» настраивал

Средний 7 мин 4.1K

Ретроспектива

+70

24

20



dmitriyrudnev

8 часов назад

Одной K155ЛА3 недостаточно

Средний 8 мин 3.5K

+53

33

5



JustJeremy

5 часов назад

«Чистый» код, ужасная производительность

16 мин 9.5K

Перевод

+40

55

75



tgeshka

16 часов назад

Хорошие, мощные и миниатюрные: mini-PC апреля. Модели для решения разных задач

3 мин 11K

+37

25

10



bodyawm

9 часов назад

Сам себе экосистема: Как я адаптировал старый смартфон под современные реалии и написал клиенты нужных мне сервисов

🔊 Средний ⌚ 11 мин 👁 3.5K

Кейс

💎 +35

📖 23

💬 24

Рoadmap от Лаборатории Касперского: какие навыки и знания нужны разработчику на C++

Спецпроект

Показать еще

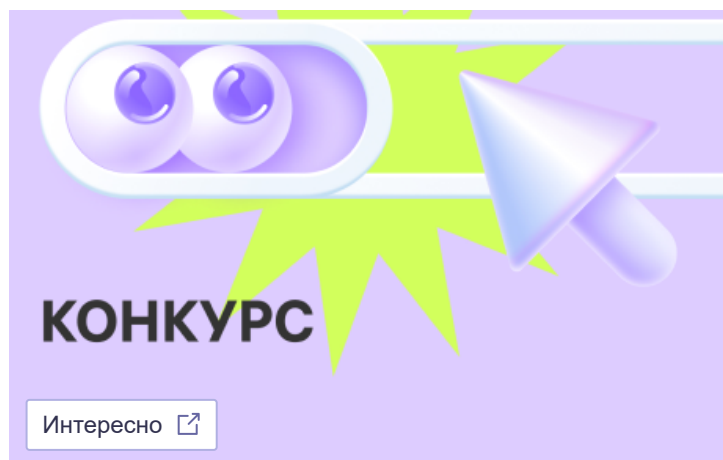
МИНУТОЧКУ ВНИМАНИЯ

Разместить



Турбо

Сезон ML — всё, но ChatGPT с нами навсегда



Получаем хабрамерч за разговоры о работе

ВОПРОСЫ И ОТВЕТЫ

Selenium nodejs android и парсинг?

Android · Средний · 0 ответов

Как ревёрс инженерить андроид приложения?

Android · Простой · 2 ответа

Почему неверно отображается текст в TextView (Java Android)?

Java · Простой · 1 ответ


Как на телевизоре HARTENS с Яндекс.TV OS сделать автозапуск HDMI при включении?


Android · Сложный · 0 ответов

Как сделать добавление в гесусcler view нового элемента без обновления списка?

Android · Простой · 1 ответ
Больше вопросов на Хабр Q&A

Реклама

 1cloud.ru РЕКЛАМА

**1cloud**

Удаленный
VDS/VPS-сервер
в аренду
за 2 минуты!

Подключить

Удаленный сервер в аренду. Бесплатное тестирование

Система виртуализации Enterprise-
уровня. SLA 99,96%. Бесплатный
мониторинг. ТП 24/7

SSL-сертификаты

>

Виртуальный ЦОД

>

Облачный сервер за 2 минуты

>

Партнёрские лицензии

>

Узнать больше

Ваш аккаунт	Разделы	Информация	Услуги
Войти	Статьи	Устройство сайта	Корпоративный блог
Регистрация	Новости	Для авторов	Медийная реклама
	Хабы	Для компаний	Нативные проекты
	Компании	Документы	Образовательные программы
	Авторы	Соглашение	Стартапам
	Песочница	Конфиденциальность	Спецпроекты



Настройка языка

Техническая поддержка

Вернуться на старую версию

© 2006–2023, Habr

Сортировка

Промпт-инженер: какие навыки освоить, чтобы зарабатывать на ChatGPT

1.1K 2

Вжух и денег нет: как Binance обнулil мой счет

50K 276

Почему ближняя к нам сторона Луны так отличается от обратной?

5K 8

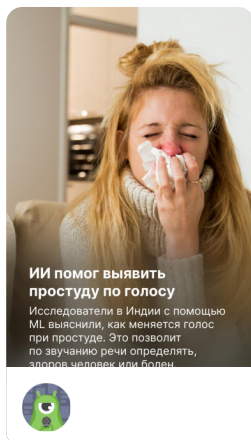
Рoadmap от Лаборатории Касперского: какие навыки и знания нужны разработчику на C++

Спецпроект

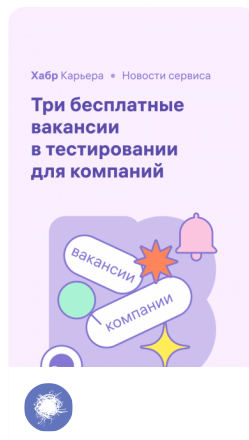
ИСТОРИИ



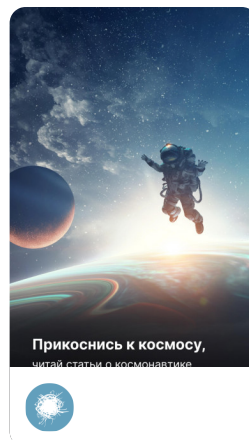
Активность найма на IT-рынке в марте



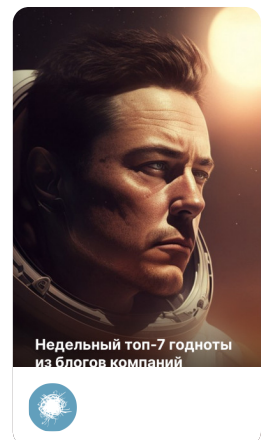
ИИ помог выявить простуду по голосу



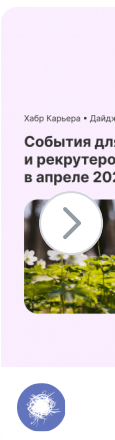
Бесплатные вакансии в тестировании



День космонавтики



Топ-7 годноты из блогов компаний



Дайджест для IT HR

РАБОТА

Swift разработчик
55 вакансий


iOS разработчик
43 вакансии

Android разработчик
35 вакансий

Все вакансии


Реклама

РЕКЛАМА

1cloud

Удаленный
VDS/VPS-сервер
в аренду
за 2 минуты!

Подключить



Удаленный сервер
в аренду. Бесплатное
тестирование

