Добавляем комфорта в работу с WebView

Тимур Гасымов

- Team Lead
- 7 лет опыта разработки под Android
- Год разработки под iOS
- 5 лет жизни с WebView

***


JS

***

Натив(ный) мобильный / Android / iOS, не NDK

В докладе не будет

Документации по WebView

Подробностей про безопасность

Документации по JavaScript

Дзен – это платформа для создателей контента

где авторы создают публикации, набирают аудиторию и зарабатывают

19 млн

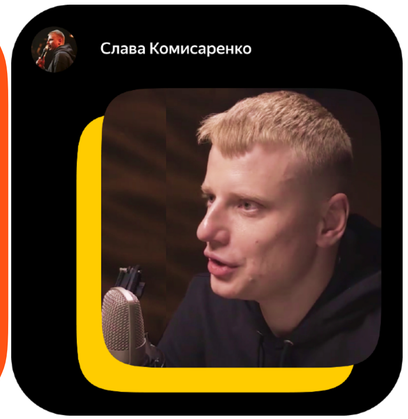
человек каждый день открывают Дзен

58 млн

месячная аудитория

45

минут пользователь проводит в Дзене в среднем за день



Дзен

300 разработчиков

400 сотрудников

2 офиса

Веб-сайт, Android и iOS

Эксперименты с реализацией UI

Используемые у нас технологии

Нативные вьюшки

Server-Driven-UI

WebView

— Почему бывает нужно использовать WebView?

~~Желание усложнить себе работу~~ НЕТ

Кросс-платформа

- Единообразный UI
- Проще обновлять и экспериментировать
- Одна разработка вместо трех для каждой платформы

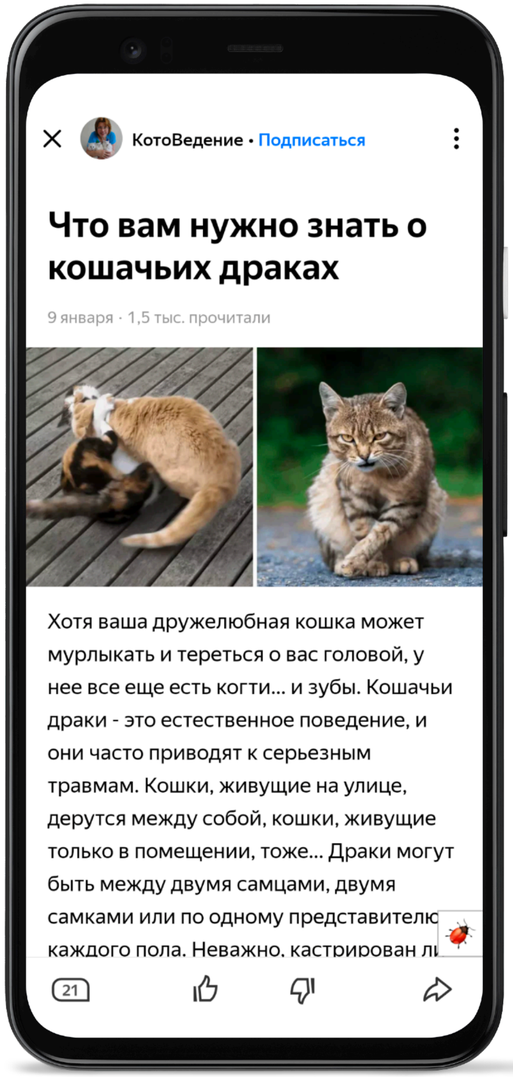
— Почему бывает нужно использовать WebView?

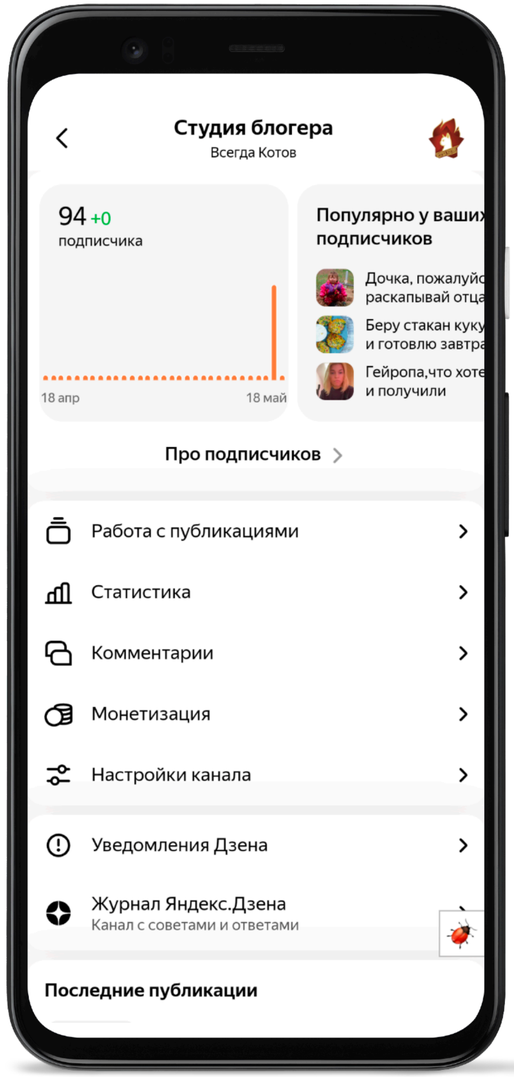
Есть уже реализованная мобильная версия сайта

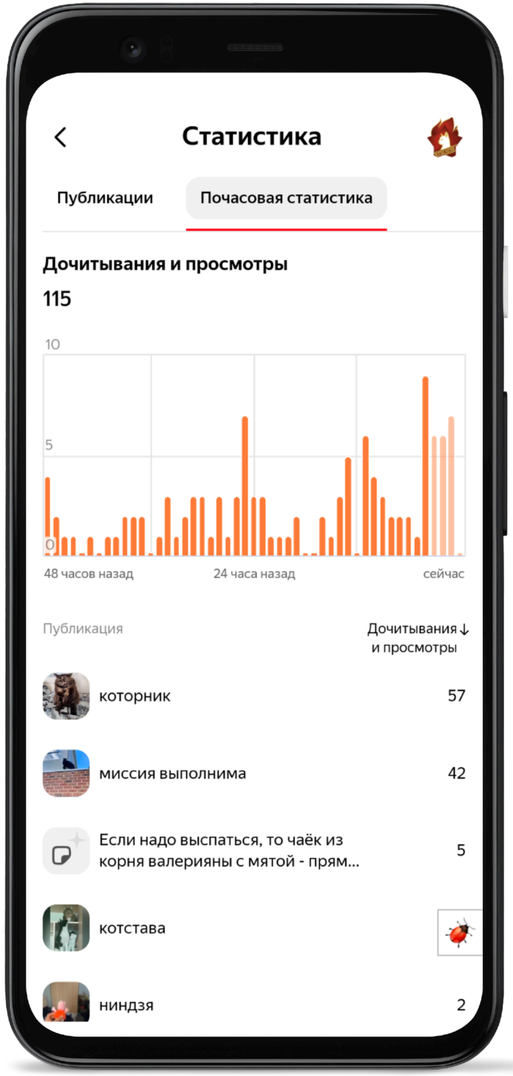
Качественно сделанный экран не портит UX

В конце доклада попробую это доказать :)

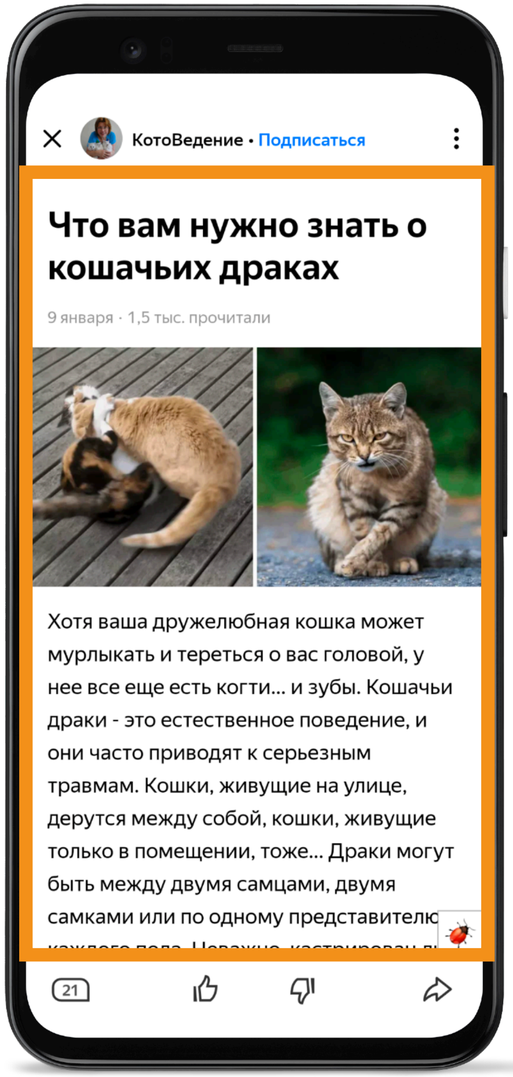
Качественные экраны с WebView в Дзене

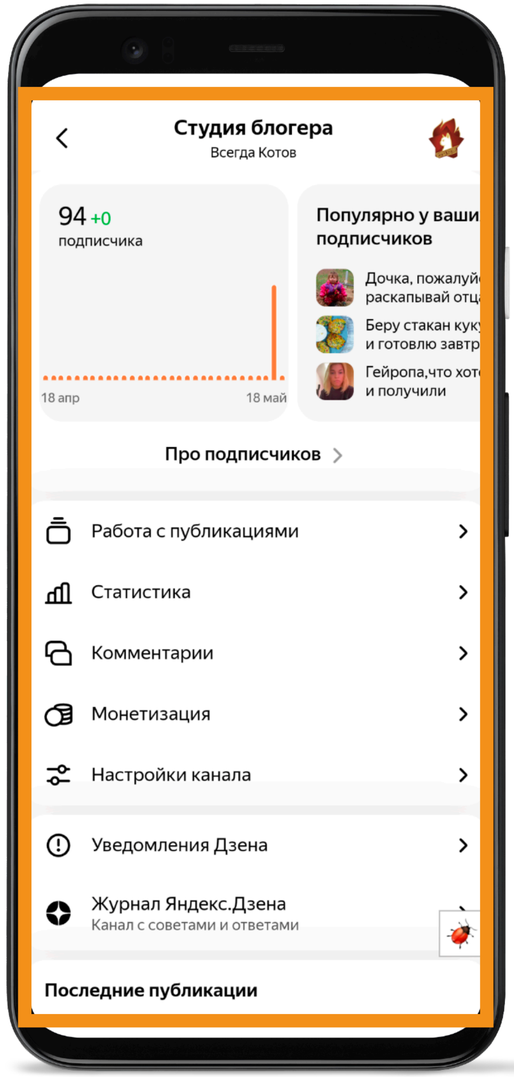


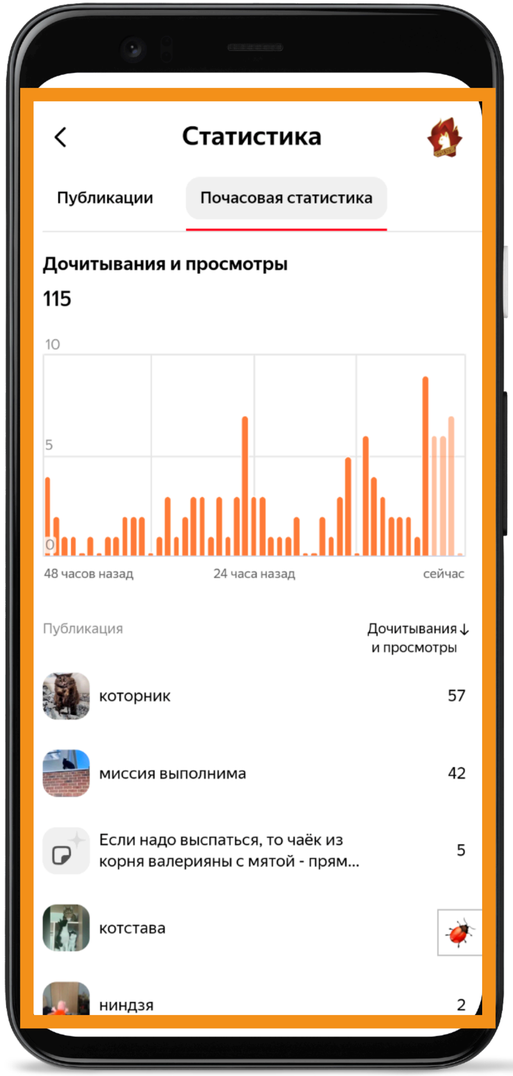




Качественные экраны с WebView в Дзене







Чего не хватает из коробки?

Нативных действий

- Авторизация
- Взаимодействие с нативными элементами
- Открытие нативных экранов

Обмена данными между веб-страницей и мобильным приложением

Как это получить

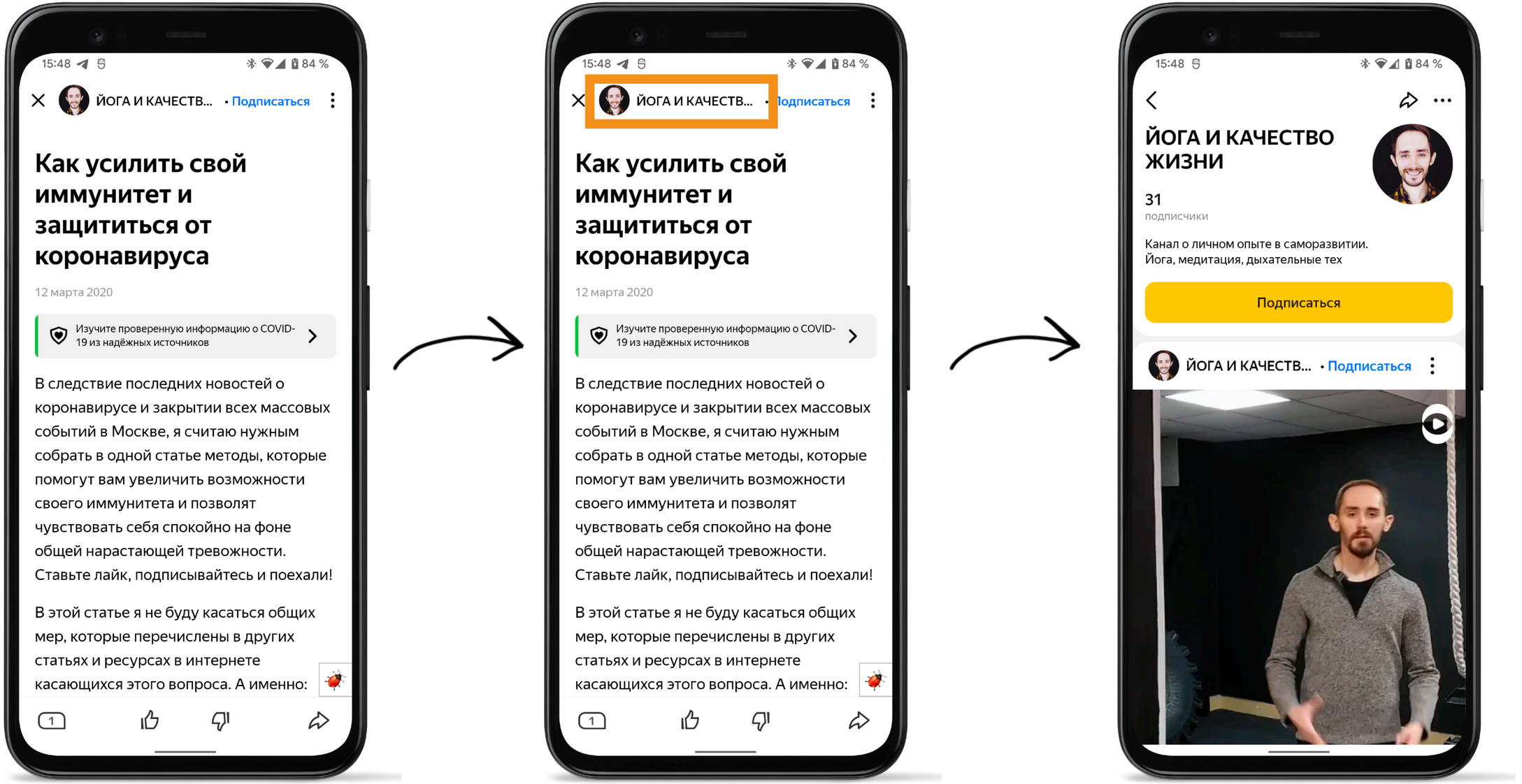
01
Ссылки с кастомной схемой

02
Коробочный @JavascriptInterface

03
Дополнительный JsBridge

— 01. Ссылки с кастомной схемой

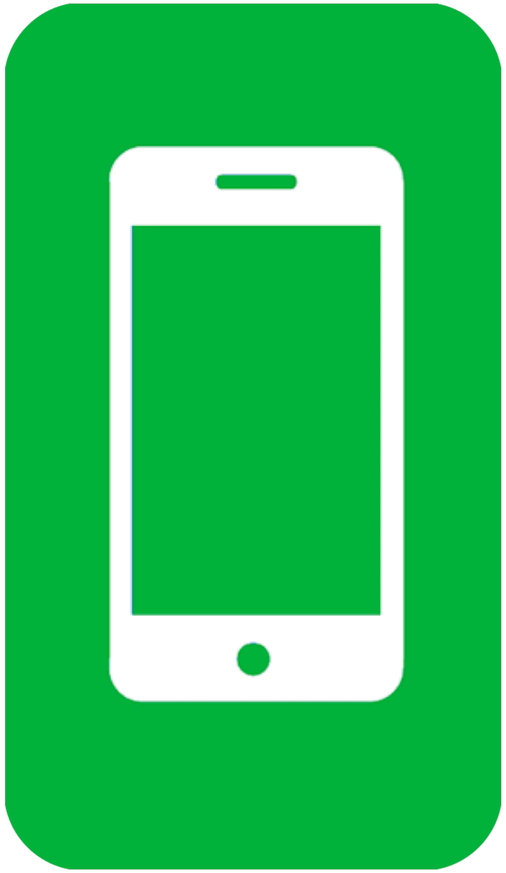
Задача №1


15:48
84 %
ЙОГА И КАЧЕСТВ...
Подписаться
Как усилить свой иммунитет и защититься от коронавируса
12 марта 2020
Изучите проверенную информацию о COVID-19 из надёжных источников
В следствие последних новостей о коронавирусе и закрытии всех массовых событий в Москве, я считаю нужным собрать в одной статье методы, которые помогут вам увеличить возможности своего иммунитета и позволят чувствовать себя спокойно на фоне общей нарастающей тревожности. Ставьте лайк, подписывайтесь и поехали!
В этой статье я не буду касаться общих мер, которые перечислены в других статьях и ресурсах в интернете касающихся этого вопроса. А именно:
1
15:48
84 %
ЙОГА И КАЧЕСТВ...
Подписаться
Как усилить свой иммунитет и защититься от коронавируса
12 марта 2020
Изучите проверенную информацию о COVID-19 из надёжных источников
В следствие последних новостей о коронавирусе и закрытии всех массовых событий в Москве, я считаю нужным собрать в одной статье методы, которые помогут вам увеличить возможности своего иммунитета и позволят чувствовать себя спокойно на фоне общей нарастающей тревожности. Ставьте лайк, подписывайтесь и поехали!
В этой статье я не буду касаться общих мер, которые перечислены в других статьях и ресурсах в интернете касающихся этого вопроса. А именно:
1
15:48
84 %
ЙОГА И КАЧЕСТВО ЖИЗНИ
31
подписчики
Канал о личном опыте в саморазвитии. Йога, медитация, дыхательные тех
Подписаться
ЙОГА И КАЧЕСТВ...
Подписаться


```
window.open("zen://navigate?to=channel");
```


```
window.open("zen://navigate?to=channel");
```

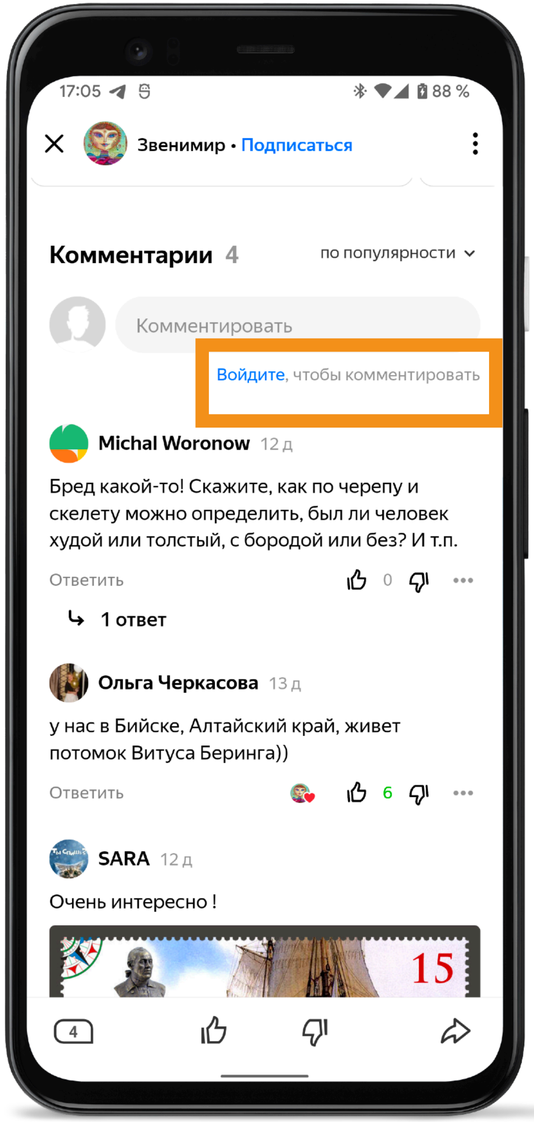
```
fun onNavigate() {
    if (scheme == "zen") {
        // ...
    }
}
```

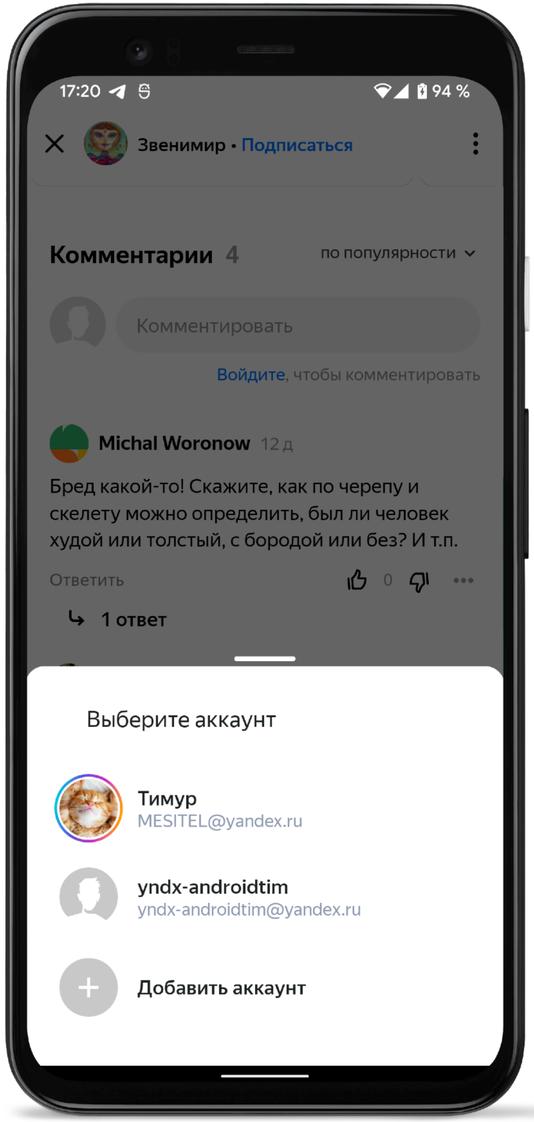


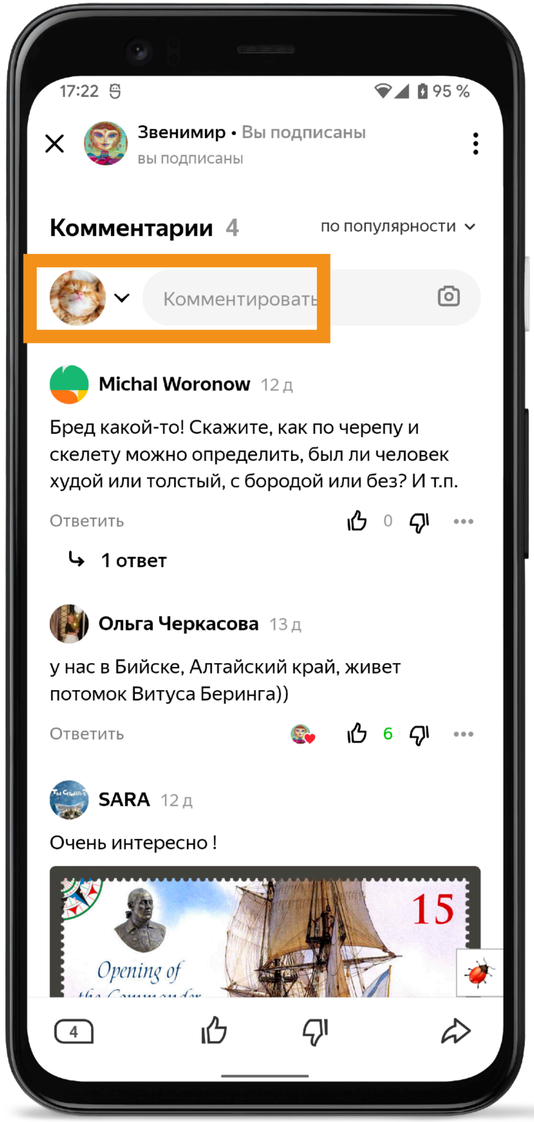
zen://navigate

— 02. Коробочный @JavascriptInterface

Задача №2


17:05
88 %
Звенимир • Подписаться
Комментарии 4
по популярности
Комментировать
Войдите, чтобы комментировать
Michal Woronow 12 д
Бред какой-то! Скажите, как по черепу и скелету можно определить, был ли человек худой или толстый, с бородой или без? И т.п.
Ответить
0
1 ответ
Ольга Черкасова 13 д
у нас в Бийске, Алтайский край, живет потомок Витуса Беринга))
Ответить
6
SARA 12 д
Очень интересно !
15
4

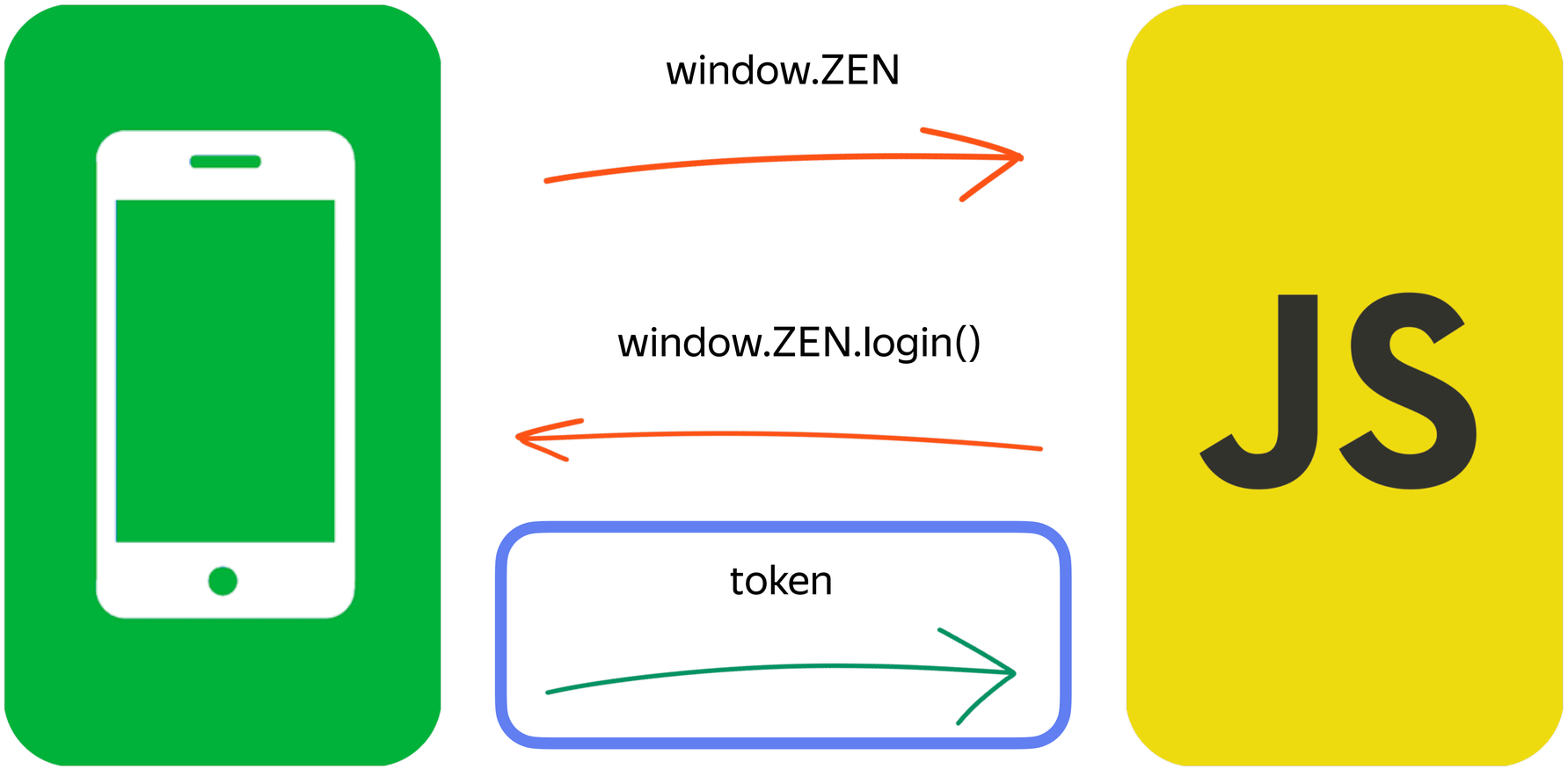

17:20
94 %
Звенимир • Подписаться
Комментарии 4
по популярности
Комментировать
Войдите, чтобы комментировать
Michal Woronow 12 д
Бред какой-то! Скажите, как по черепу и скелету можно определить, был ли человек худой или толстый, с бородой или без? И т.п.
Ответить
0
1 ответ
Выберите аккаунт
Тимур
MESITEL@yandex.ru
yndx-androidtim
yndx-androidtim@yandex.ru
Добавить аккаунт


17:22
95 %
Звенимир • Вы подписаны
вы подписаны
Комментарии 4
по популярности
Комментировать
Michal Woronow 12 д
Бред какой-то! Скажите, как по черепу и скелету можно определить, был ли человек худой или толстый, с бородой или без? И т.п.
Ответить
0
1 ответ
Ольга Черкасова 13 д
у нас в Бийске, Алтайский край, живет потомок Витуса Беринга))
Ответить
6
SARA 12 д
Очень интересно !
15
Opening of
4

```
webView.addJavaScriptInterface(ZenJsApi(), "ZEN")
```

```
webView.addJavaScriptInterface(ZenJsApi(), "ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        // ...
    }
}
```

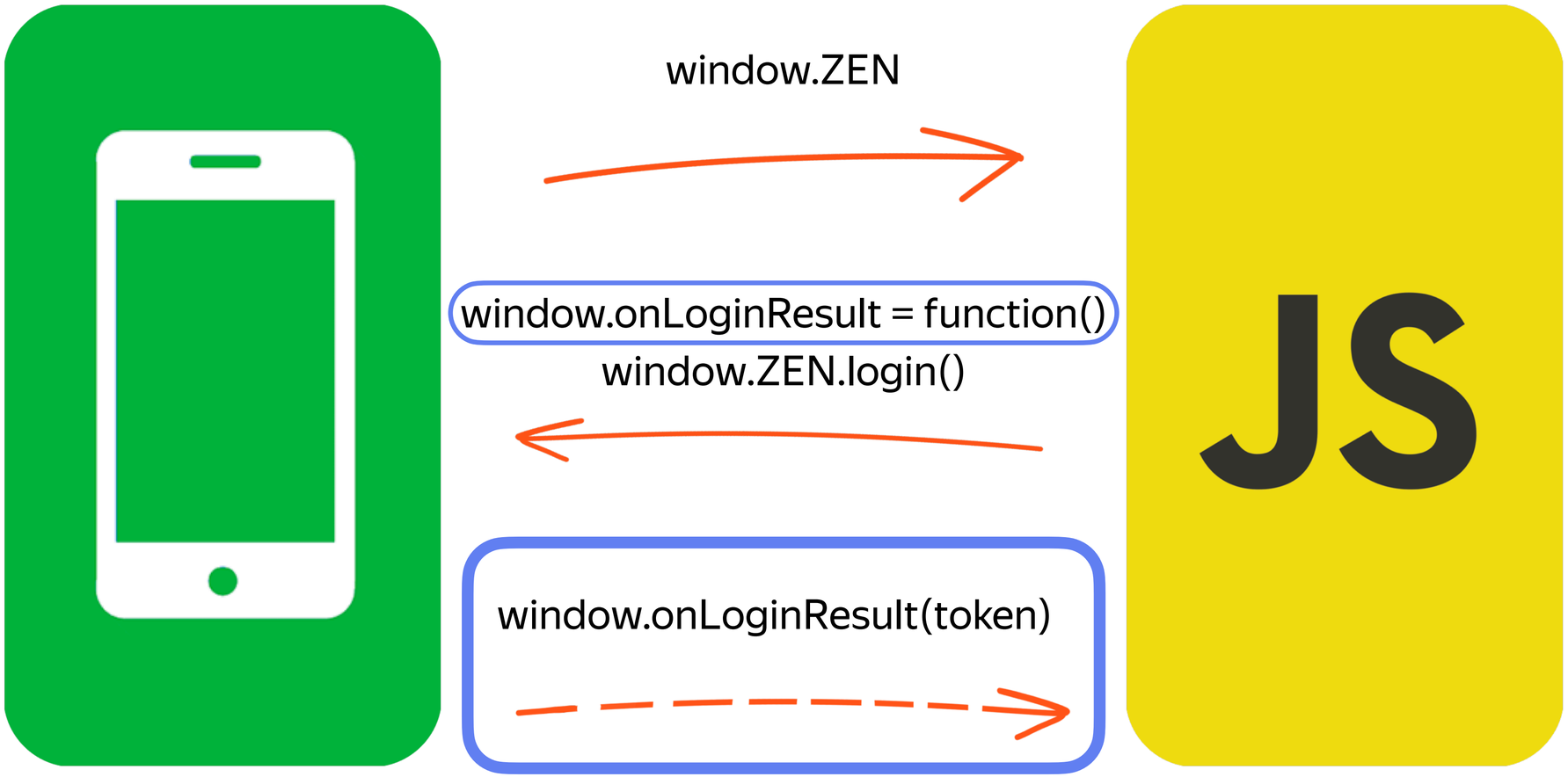

```
const token = window.ZEN.login();
```


window.ZEN
window.ZEN.login()
token
JS


```
window.onLoginResult = (token) => {
    // ...
};
```

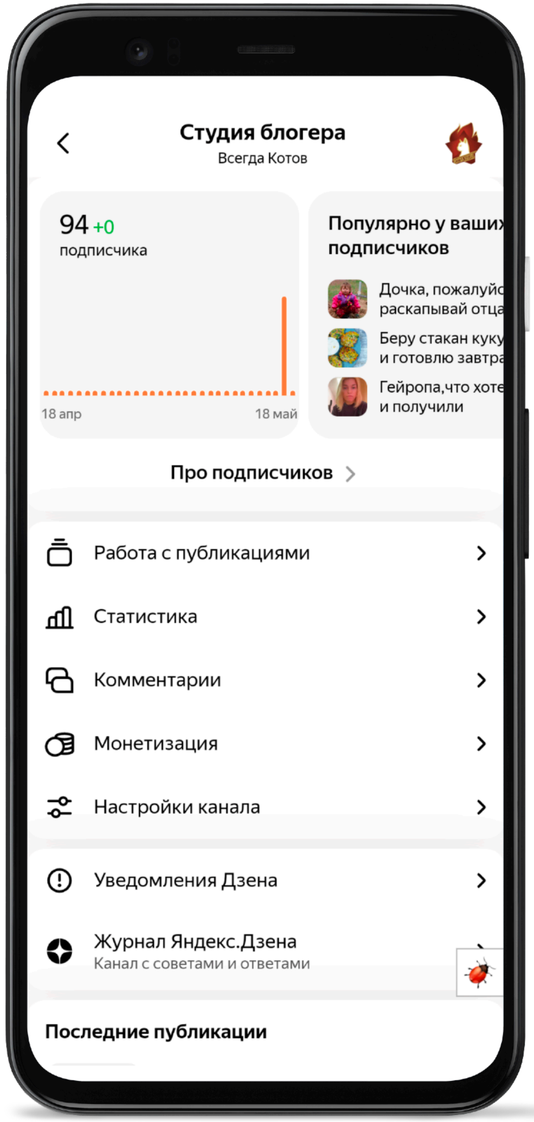
```
window.ZEN.login();
```

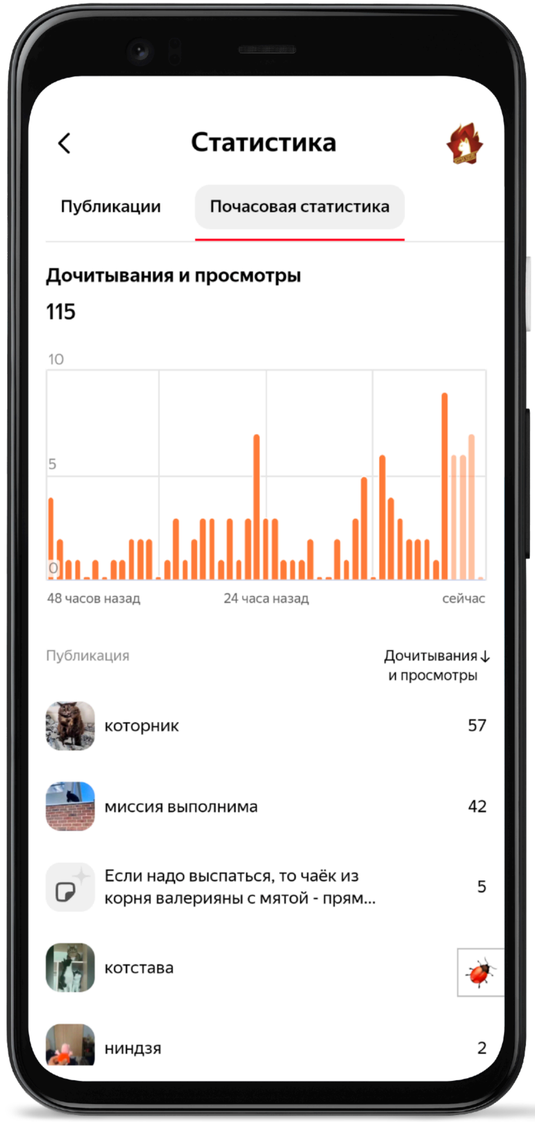
```
webView.addJavaScriptInterface(ZenJsApi(), "ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        // ...
        onLogin { token ->
            webView.evaluateJavaScript("window.onLoginResult(${token})")
        }
    }
}
```

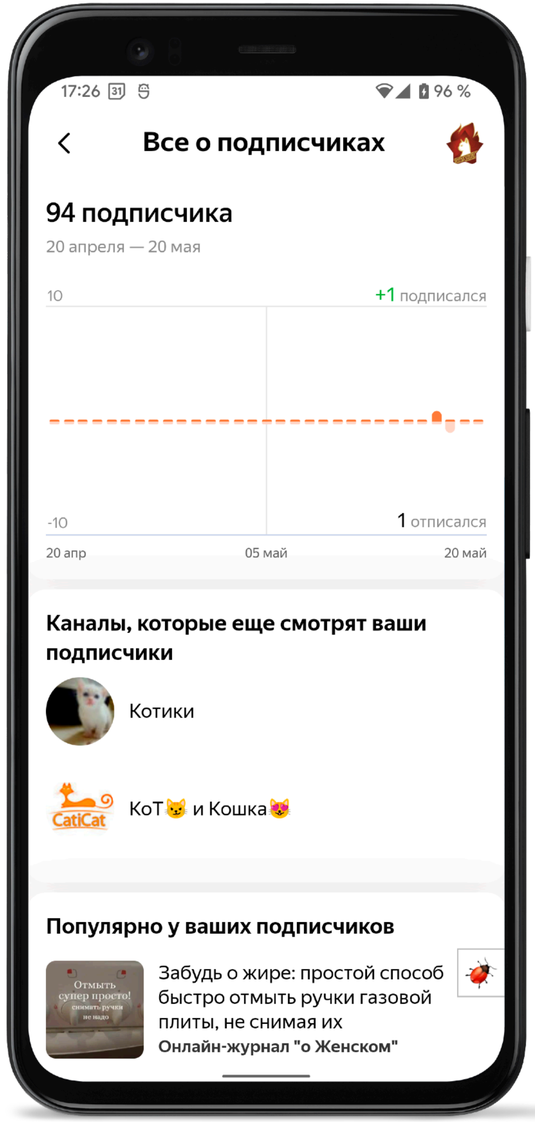

window.ZEN
window.onLoginResult = function()
window.ZEN.login()
window.onLoginResult(token)
JS

— 03. Дополнительный JsBridge

Задача №3


Студия блогера
Всегда Котов
94 +0
подписчика
18 апр
18 май
Популярно у ваших подписчиков
Дочка, пожалуйс раскапывай отца
Беру стакан куку и готовлю завтра
Гейропа,что хоте и получили
Про подписчиков
Работа с публикациями
Статистика
Комментарии
Монетизация
Настройки канала
Уведомления Дзена
Журнал Яндекс.Дзена
Канал с советами и ответами
Последние публикации


Статистика
Публикации
Почасовая статистика
Дочитывания и просмотры
115
10
5
0
48 часов назад
24 часа назад
сейчас
Публикация
Дочитывания и просмотры
которник 57
миссия выполнима 42
Если надо выспаться, то чаёк из корня валерияны с мятой - прям... 5
котстава
ниндзя 2


17:26
96 %
Все о подписчиках
94 подписчика
20 апреля — 20 мая
10
+1 подписался
-10
1 отписался
20 апр
05 май
20 май
Каналы, которые еще смотрят ваши подписчики
Котики
КоТ🐱 и Кошка😻
Популярно у ваших подписчиков
Забудь о жире: простой способ быстро отмыть ручки газовой плиты, не снимая их
Онлайн-журнал "о Женском"

```ts
interface JSAPI {
    // закрывает страницу
    ) => Promise<void>;

    // Оборачивание переданного урла в редирект паспорта, который выставит куки
    ing): Promise<string>

    // Нотификация о том, что изменился статус подписки на канал. Статистику на сервер должен отправить JS (веб-страница)
    : Promise<void>;
    ): Promise<void>; // нотификация о том, что данные внутри webview готовы к показу
    ): Promise<void>; // нотификация о том, что внутри webview случилась ошибка, надо показать нативный контрол рефреша

    ): Promise<void>;  // Будет reject с ошибкой, если что-то пошло не так
    // Открытие публикации с платформы
    // requireUserData - выставление авторизационных хэдеров на открываемой странице
    : Promise<void>;
    // Открытие нативного экрана канала. Тип Channel — см. ниже
    // Для открытия чужого профиля нужно ориентироваться на присутствие поля link в Channel
    : Promise<void>;
    // Экран (таб) со списком подписок
    : Promise<void>;
    // Экран настроек
    ): Promise<void>;
    // Экран истории
    : Promise<void>;
    // Экран веб-вью мобильного редактора
    Promise<void>;
    // Экран веб-вью редактора постов
    ): Promise<void>;
    // Экран нотификаций
    : Promise<void>;
    // Экран своего профиля (для возврата с чужих подписок например
    // параметр reset определяет сбрасывать ли стек навигации
    ): Promise<void>;
    // Открытие нативного канала автора. Тип Channel — см. ниже
    // Чтобы автор мог сразу увидеть свой канал и публикации
    ): Promise<void>;

```


```
try {
    const result = await window.ZEN.navigate(arg);
} catch(err) {
    console.log("Error: $err");
}
```

```
const promises = {};
const zen = window.NATIVE_ZEN;

for (let methodName in zen) {
  let fn = zen[methodName];

  ZEN[methodName] = () => {
    let args = arguments;
    let promiseId = guid();
    return new Promise((resolve, reject) => {
      promises[promiseId] = {
        resolve: resolve,
        reject: reject
      };

      try {
        var processedArgs = Array.from(args).map((arg) => {
          return typeof arg === 'object' ? JSON.stringify(arg) : arg;
        });
        fn.apply(zen, [promiseId].concat(Array.from(processedArgs)));
      } catch (e) {
        reject(e);
      }
    });
  };
}

window.handleZenPromiseResult = (e) => {
  let promise = promises[e.promiseId];

  if (e.status === 'SUCCESS') {
    promise.resolve(e.data);
  } else {
    promise.reject(new Error(e.error));
  }
};
```

в нативе регистрируем объект с другим именем

проходим циклом по всем его методам

создаем новый объект с такими же функциями

генерируем уникальный promiseID

создаем Promise

сохраняем его

передаем в натив promiseID

регистрируем функцию для получения результата

возвращаем результат в Promise

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

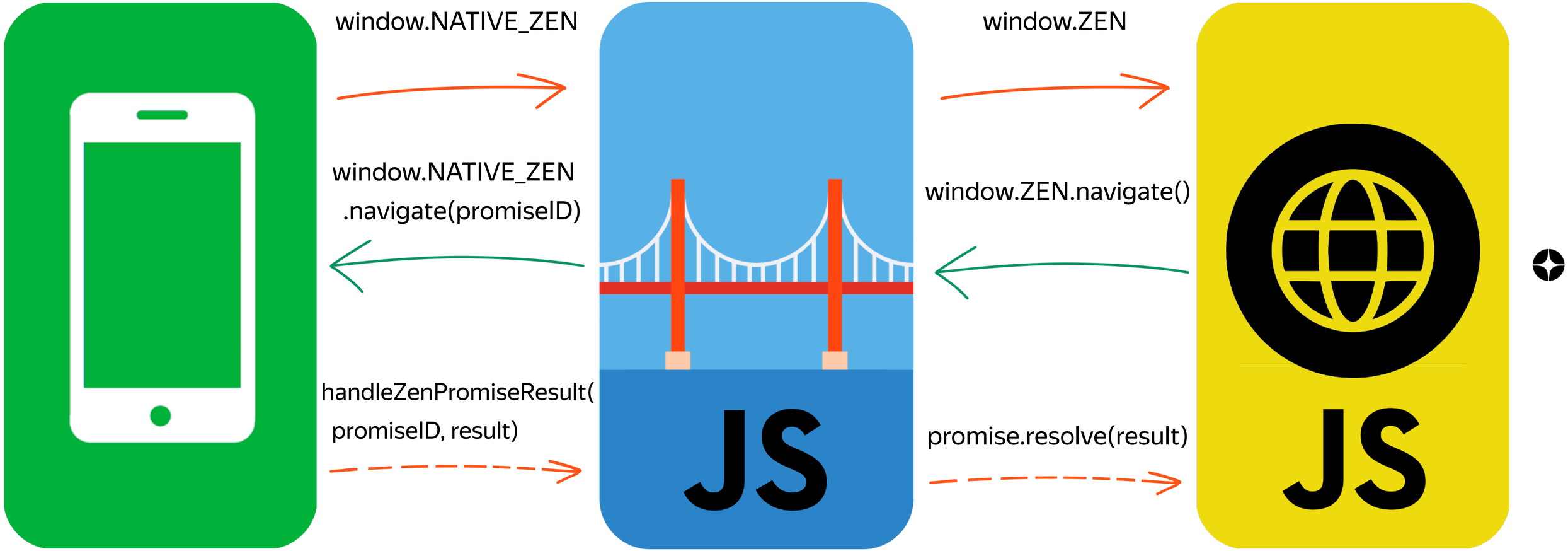
```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```

```
webView.addJavaScriptInterface(ZenJsApi(), "NATIVE_ZEN")

class ZenJsApi {
    fun onReceiveMessage(message) {
        val promiseId = message.promiseId
        // ...
        onLogin { token ->
            if (result != null) {
                sendPromiseResolve(promiseId, token)
            } else {
                sendPromiseReject(promiseId, "Error on login")
            }
        }
    }

    fun sendPromiseResolve(promiseId, result) {
        val promiseResult = asJson {
            "promiseId" to promiseId,
            "status" to SUCCESS,
            "data" to result,
        }
        webView.evaluateJavaScript("window.handleZenPromiseResult(${promiseResult})")
    }
}
```


window.NATIVE_ZEN
window.ZEN
window.NATIVE_ZEN
.navigate(promiseID)
window.ZEN.navigate()
handleZenPromiseResult(
promiseID, result)
promise.resolve(result)
JS
JS

Что дает JsBridge

- Мобильным разработчикам проще расширять API
- Красивый и привычный код для веб-разработчиков
- Позволяют делать несколько асинхронных вызовов одной функции
- Предоставляют больше информации в случае ошибки
- Дополнительный контроль и безопасность
- Автоматическая сериализация JS-объектов


Remember, with great power
comes great responsibility.

Что дает JsBridge

- Мобильным разработчикам проще расширять API
- Красивый и привычный код для веб-разработчиков
- Позволяют делать несколько асинхронных вызовов одной функции
- Предоставляют больше информации в случае ошибки
- Дополнительный контроль и безопасность
- Автоматическая сериализация JS-объектов
-  нужно доставить на веб-страницу

```
class ZenWebViewClient : WebViewClient() {
    override fun onPageStarted() {
        // 1
    }

    override fun onPageFinished() {
        // 2
    }
}
```

```
class ZenWebViewClient : WebViewClient() {
    override fun onPageStarted() {
        jsApi.inject()
    }

    override fun onPageFinished() {
        // 2
    }
}
```

```
class ZenWebViewClient : WebViewClient() {
    override fun onPageStarted() {
        jsApi.init()
    }

    override fun onPageFinished() {
        // 2
    }
}
```

```
class ZenWebViewClient : WebViewClient() {
    override fun onPageStarted() {
        // 1
    }

    override fun onPageFinished() {
        jsApi.inject()
    }
}
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```

```
const launchOptions = {};

if (!window.ZEN_LAUNCH_OPTIONS) {
    window.ZEN_LAUNCH_OPTIONS = Promise.resolve(launchOptions);
} else {
    window.ZEN_LAUNCH_OPTIONS.resolve(launchOptions);
}
```

```
let promise = window.ZEN_LAUNCH_OPTIONS;
if (!promise) {
    let resolve;
    promise = window.ZEN_LAUNCH_OPTIONS = new Promise((_resolve) => {
        resolve = _resolve;
    });
    promise.resolve = resolve;
}
const launchOptions = await promise;

window.ZEN.doAction()
```


```
<script src="local://mobile_js_api_sync_inject"></script>
```


```
<script src="local://mobile_js_api_sync_inject"></script>
<script>
    window.ZEN.updateArticleState(articleState);
</script>
```


узбагойся

Сравнение трёх подходов

	Deeplink	@JavascriptInterface	JsBridge
Выполнение действий	✓	✓	✓
Возврат результата	✗	✓	✓
Асинхронное выполнение	✗	✓	✓
Возврат ошибок	✗	✗	✓
Красивый код	✗	✗	✓
Не нужно ничего придумывать	✓	✓	✗

Чего это стоит?

Необходимость дополнительной js-обертки

Мобильным разработчикам необходимо повышать квалификацию

Сложнее отлаживать баги

В самой WebView много подводных камней


Но у вас же есть
Server-Driven-UI

Почему мы все еще используем WebView?

Самый быстрый и понятный способ реализовать срочную фичу

Открытие сторонних сайтов

На 99% кросс-платформенные экраны, с одинаковым UI и UX

Оно не страшное,
если научится его готовить :)

Советы

Делайте классный и нативный UI

Добавьте возможность использовать WebView

Думайте о том, какой инструмент использовать

Советы

Спасибо

Моя библиотека JsApiBridge

Доклад про Server-Driven-UI (DivKit)

Гасымов Тимур

tumolllaa@gmail.com

@androidtim