Яндекс

Mobius

По ту сторону EditText. Программная клавиатура в Android

Вадим Черненко, Дмитрий Дегтярёв

Содержание

01 Введение

Незаменимый помощник

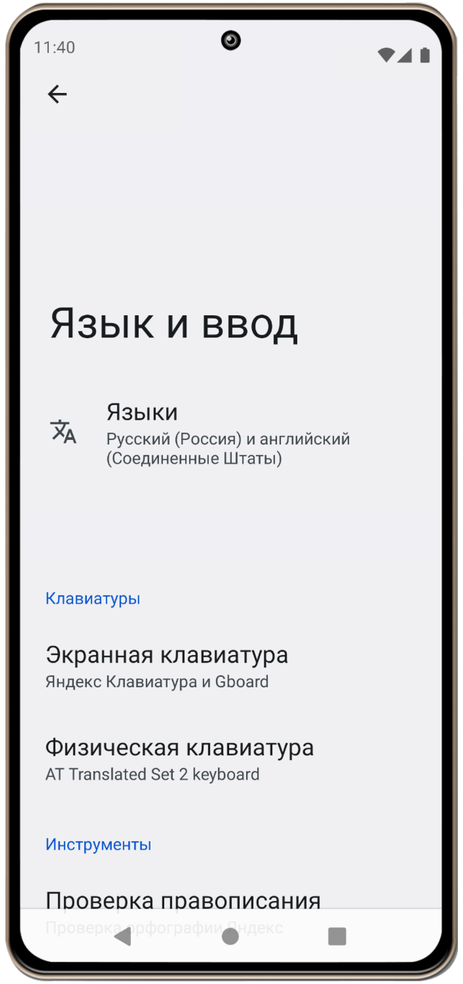
Необычное приложение

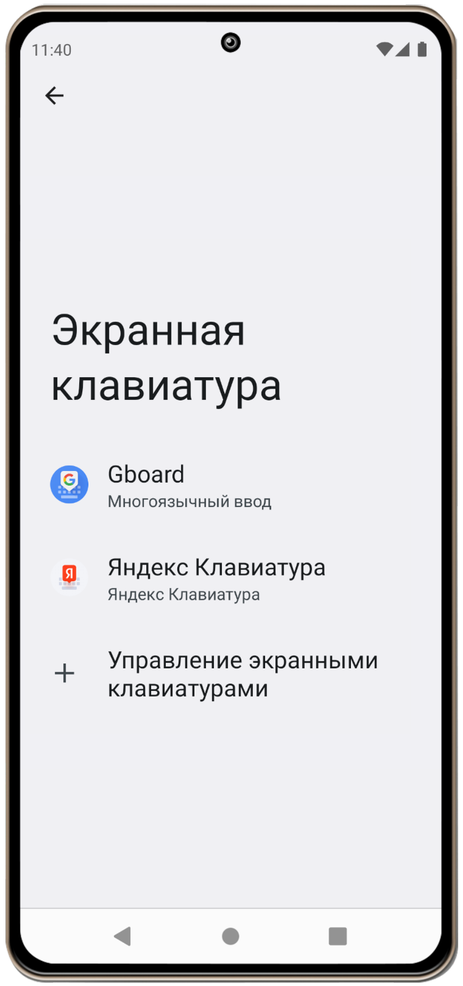
Повышенная ответственность

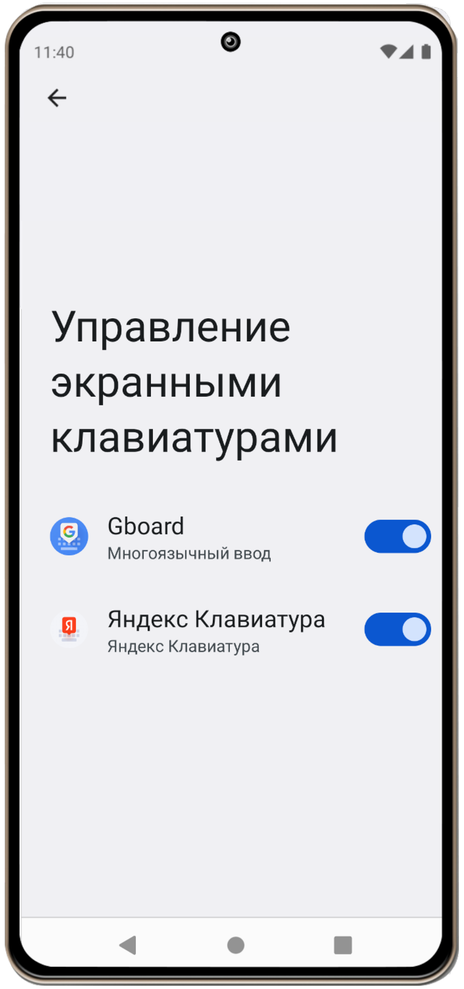
02 Прототип

Сервис ввода

- Относится к системному набору функциональных сервисов
- Обладает расширенным жизненным циклом
- Попадает под некоторые исключения из правил ОС







Создание сервиса ввода

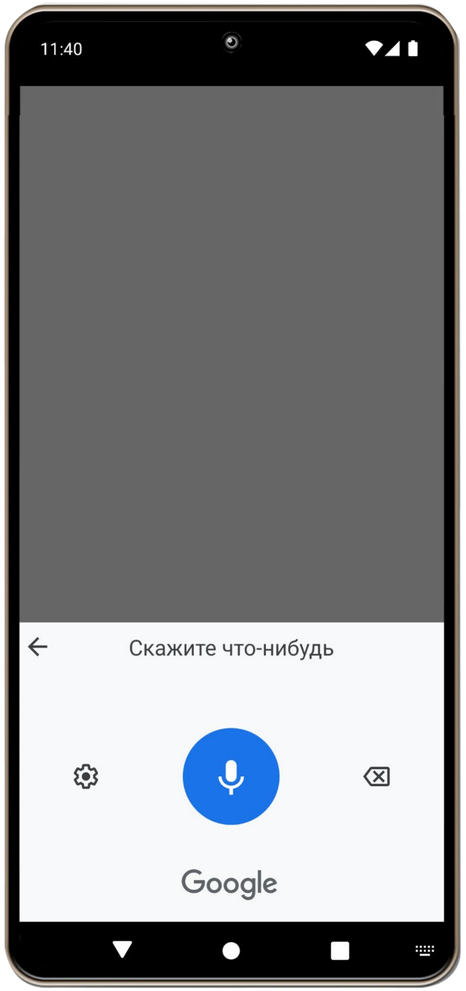
```
class ExampleInputMethodService : InputMethodService() {
    override fun onCreateInputView(): View? {
        // TODO: implement me
        return null
    }
}
```

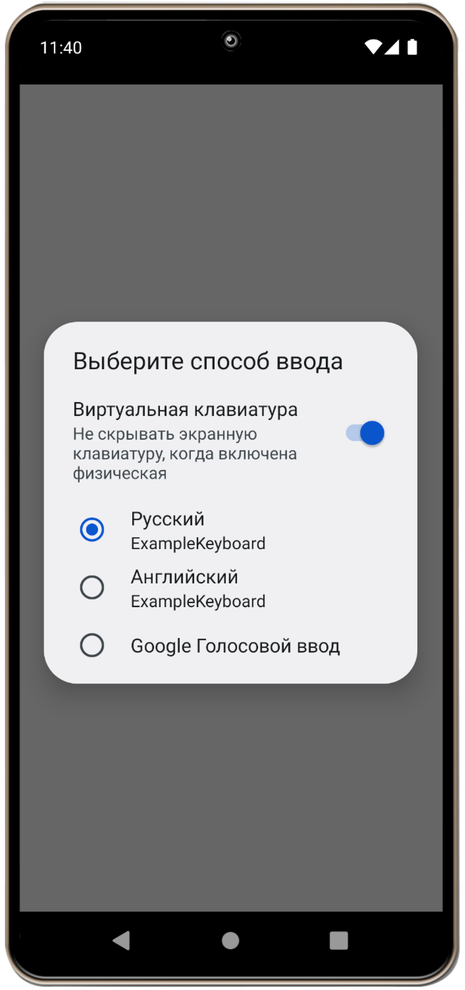
Регистрация сервиса ввода

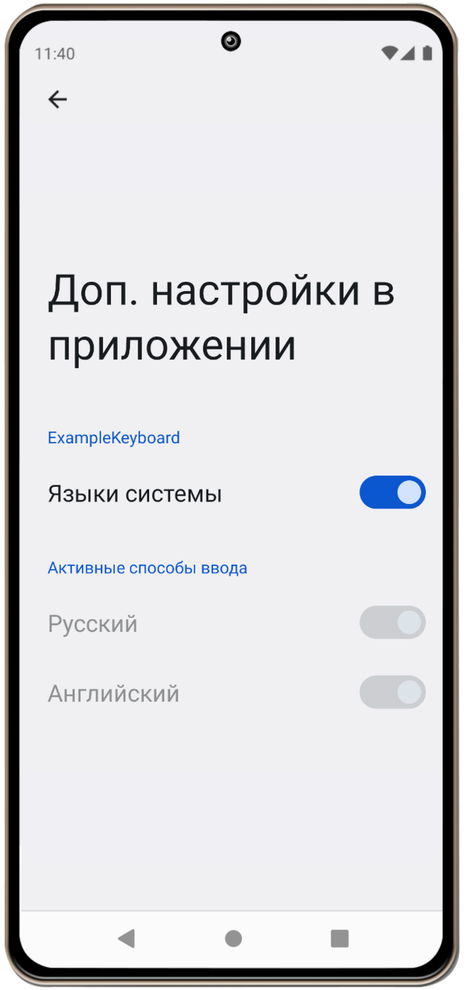
```
<service
    android:name=".ExampleInputMethodService"
    android:label="@string/input_method_label"
    android:exported="true"
    android:permission="android.permission.BIND_INPUT_METHOD">
    <meta-data android:name="android.view.im"
        android:resource="@xml/input_method_subtypes" />
    <intent-filter>
        <action android:name="android.view.InputMethod" />
    </intent-filter>
</service>
```

Сабтайпы

- Описывают локаль или тип ввода
- Представляют клавиатуру в списке доступных для включения и выбора
- Оказывают влияние на некоторые компоненты системы





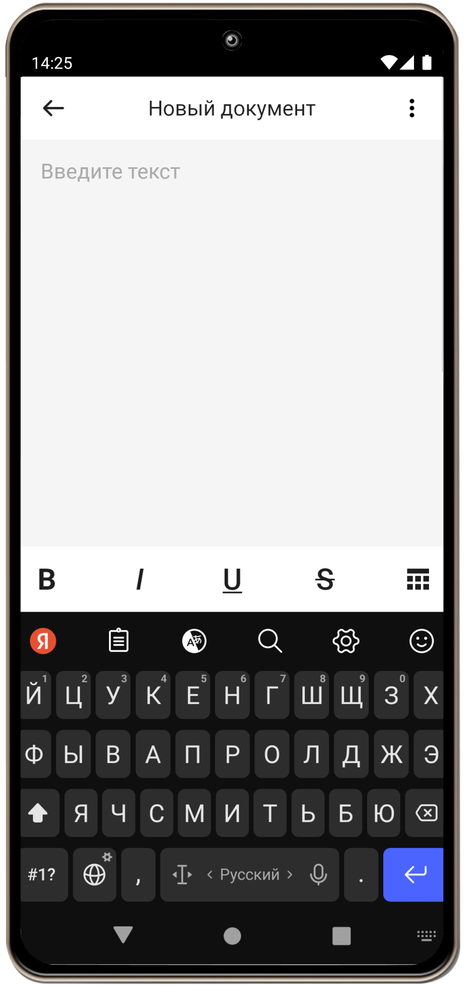


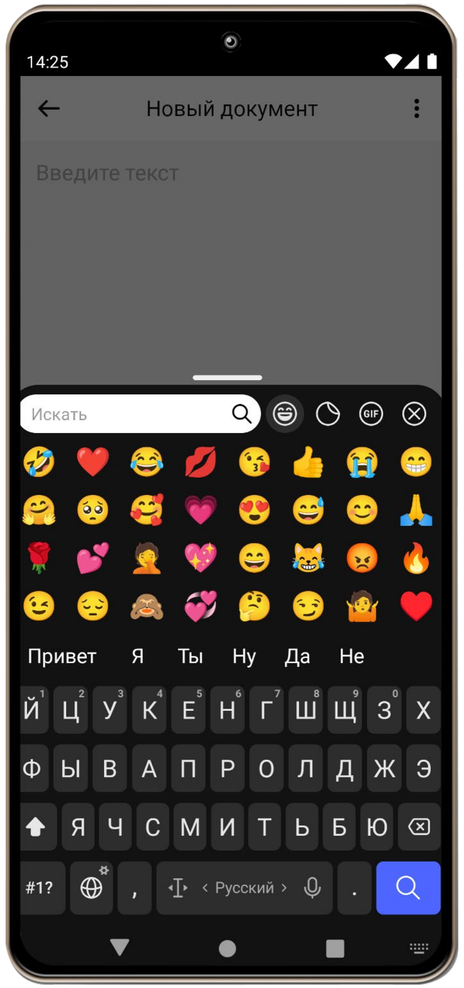
Регистрация сабтайпов

```
<input-method
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:settingsActivity="com.example.keyboard.SettingsActivity">
    <subtype
        android:label="@string/input_method_subtype_label_en"
        android:isAsciiCapable="true"
        android:imeSubtypeMode="keyboard"
        android:imeSubtypeLocale="en_US" />
    <!-- ... -->
</input-method>
```

UI клавиатуры

- View
- Свобода в реализации рендеринга и обработки событий
- Позиционирование поверх активного приложения с обработчиком ввода



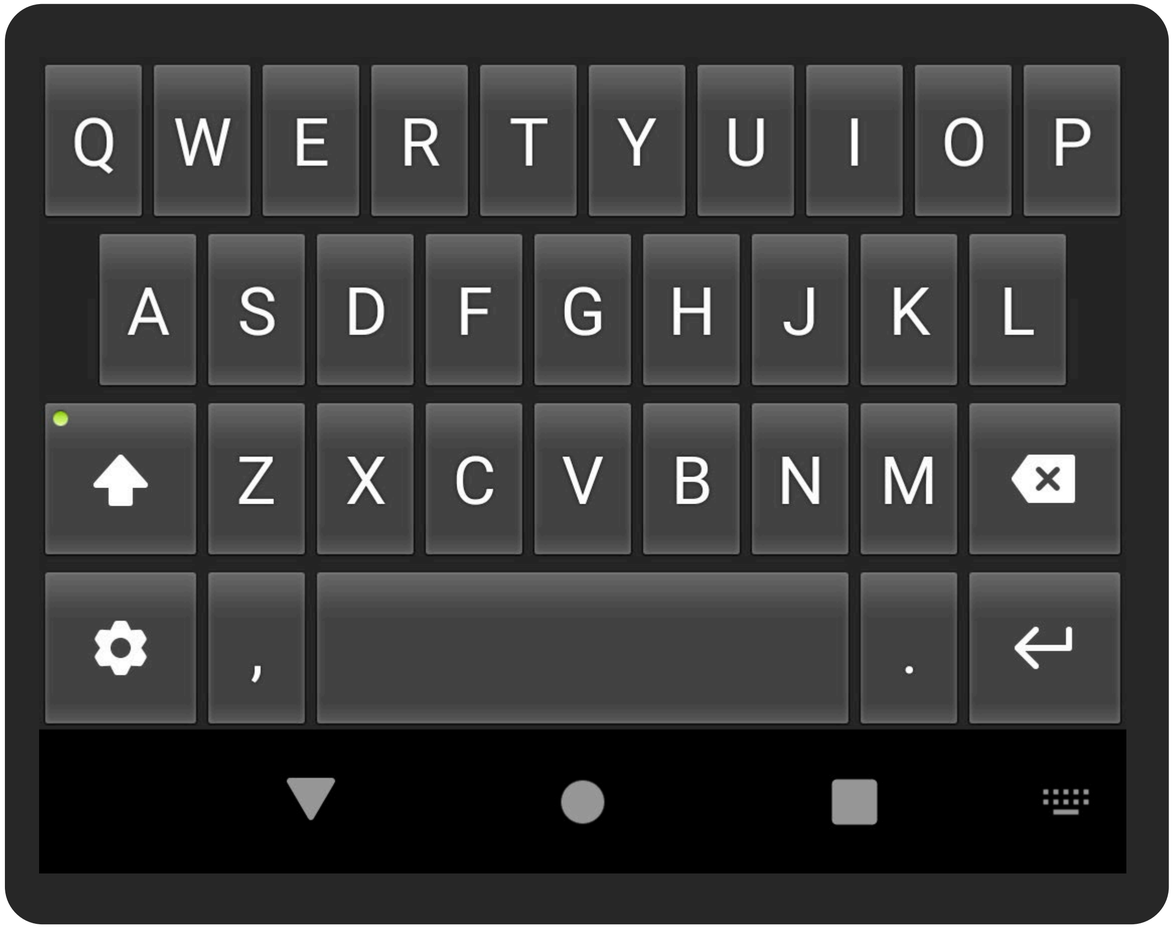


Разметка UI на основе Keyboard

```
<Keyboard
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:keyWidth="10%p"
    android:keyHeight="56dp">
    <Row android:rowEdgeFlags="top">
        <Key android:codes="113" android:keyLabel="q" android:keyEdgeFlags="left" />
        <!-- ... -->
        <Key android:codes="112" android:keyLabel="p" android:keyEdgeFlags="right" />
    </Row>
    <!-- ... -->
</Keyboard>
```

Создание UI на основе KeyboardView

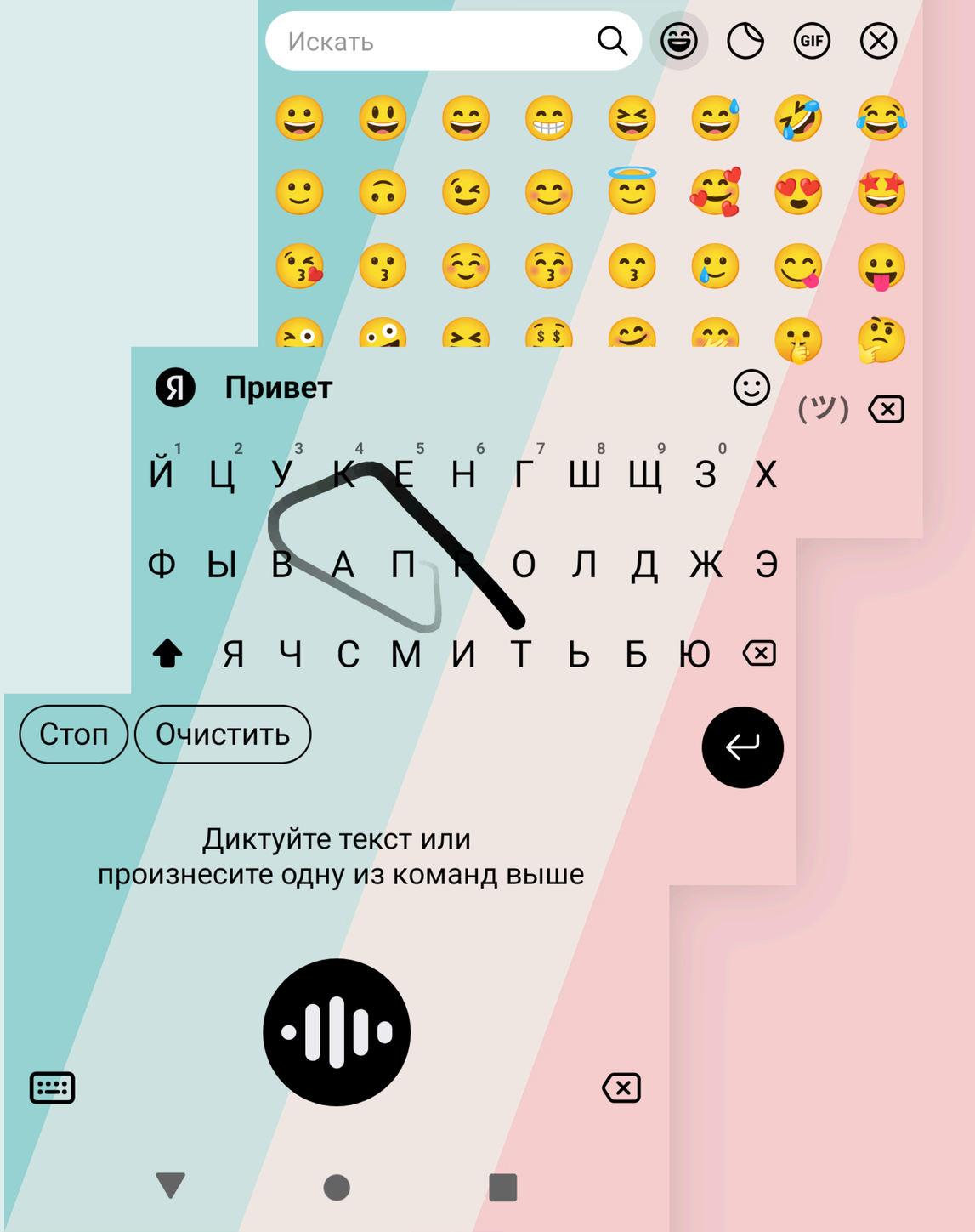
```
class ExampleInputMethodService : InputMethodService() {

    override fun onCreateInputView(): View? {

        val keyboardView = KeyboardView(this, null)

        keyboardView.keyboard = Keyboard(this, R.xml.keyboard_layout_en)

        keyboardView.setOnKeyboardActionListener(ExampleOnKeyboardActionListener {

            currentInputConnection

        })

        return keyboardView

    }

}
```


Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
,
.

Обработка пользовательского ввода

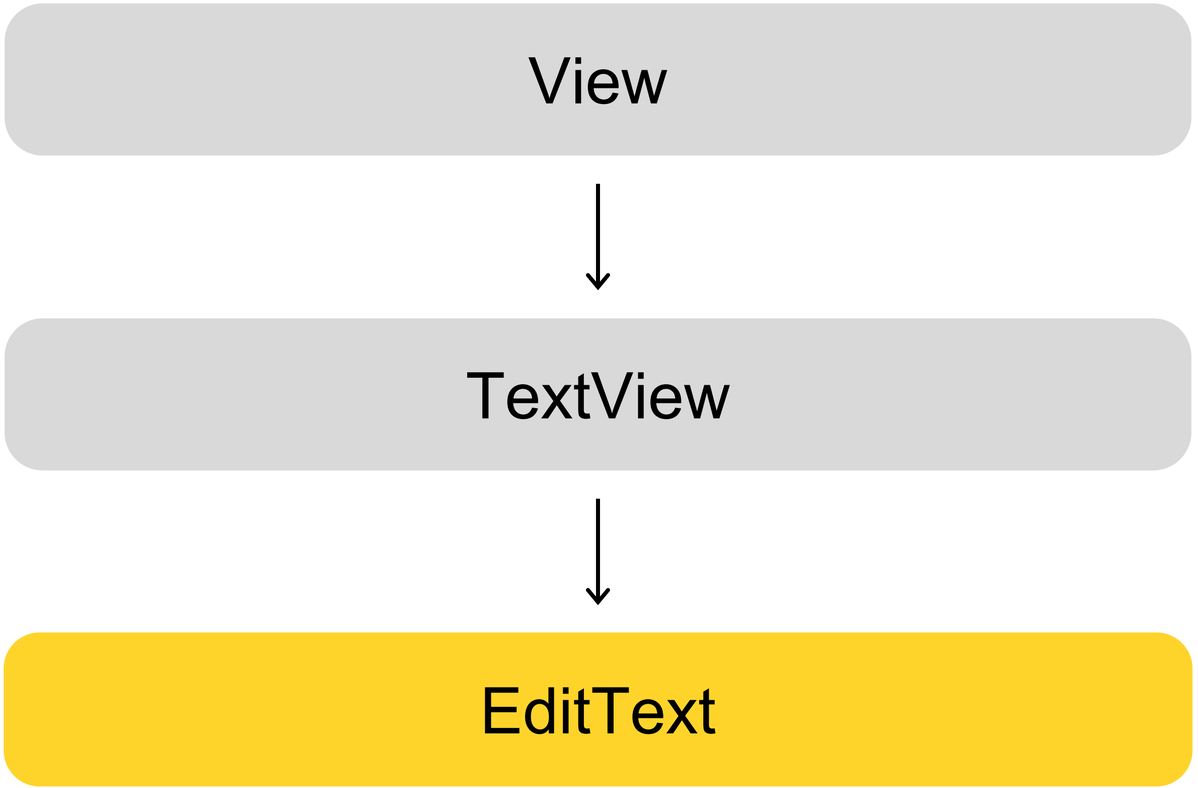
```
class ExampleOnKeyboardActionListener : OnKeyboardActionListenerStub() {
    override fun onKey(primaryCode: Int, keyCodes: IntArray) {
        when (primaryCode) {
            // TODO: handle special keys
            else -> {
                val text = primaryCode.toChar().toString()
                inputConnectionProvider()?.commitText(text, 1)
            }
        }
    }
}
```

Яндекс Клавиатура

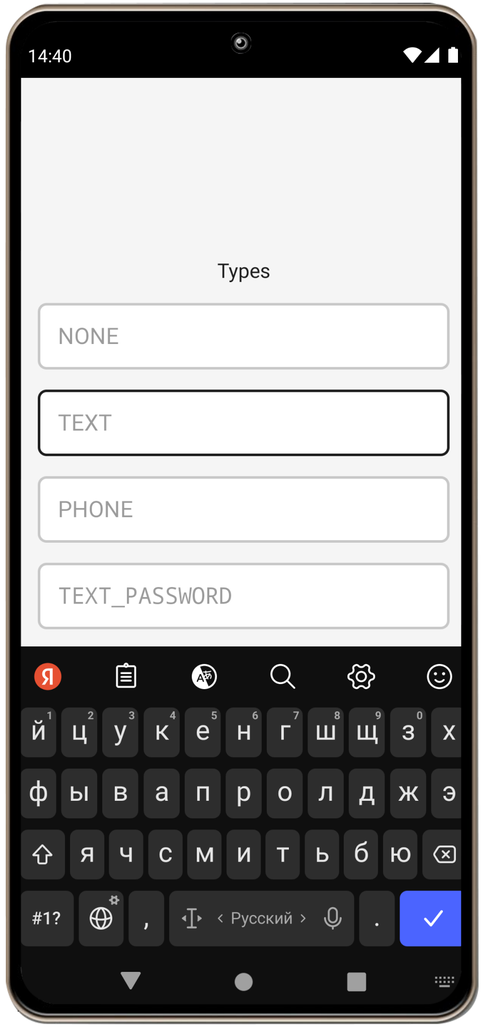


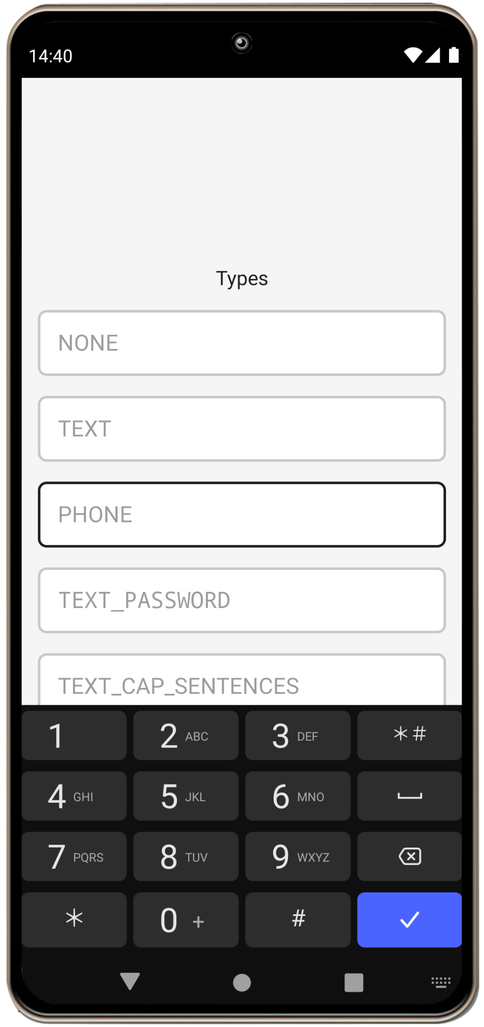
03 Великолепная четвёрка

EditText

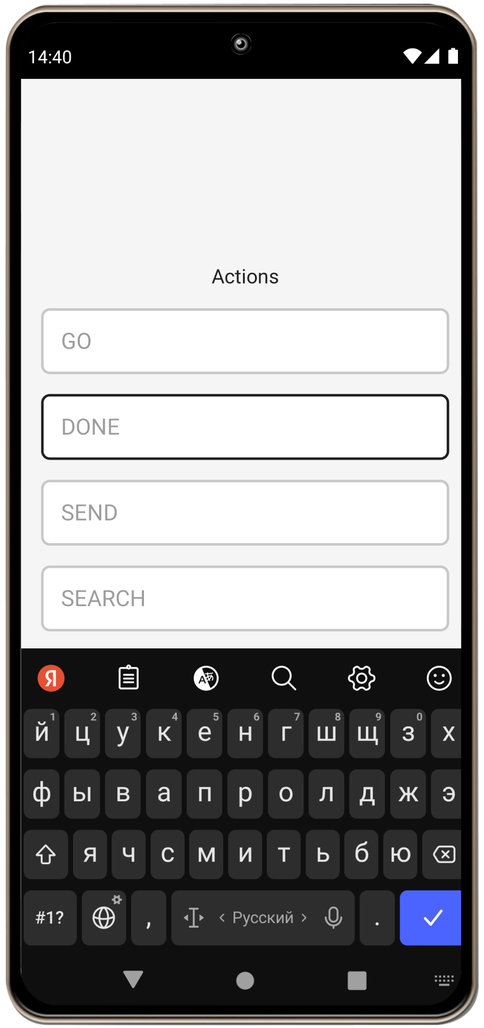


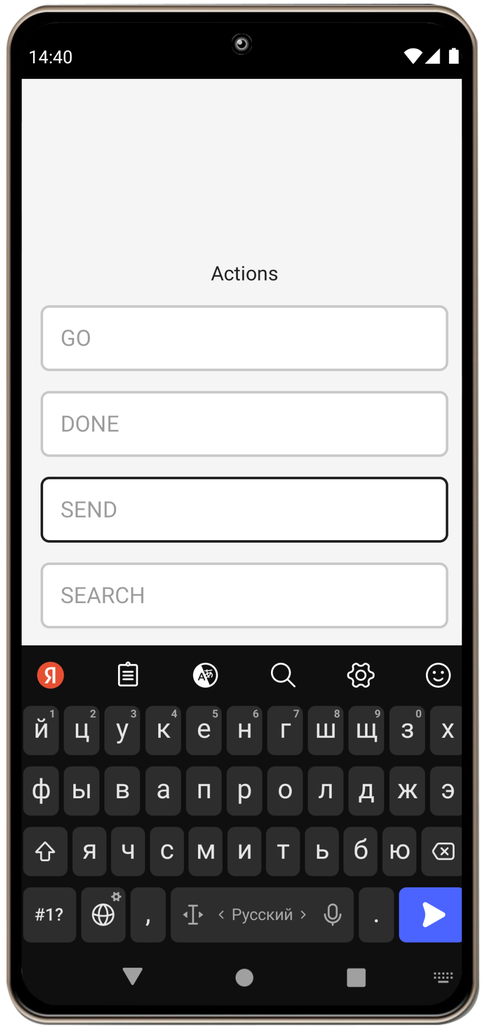
Атрибут inputType

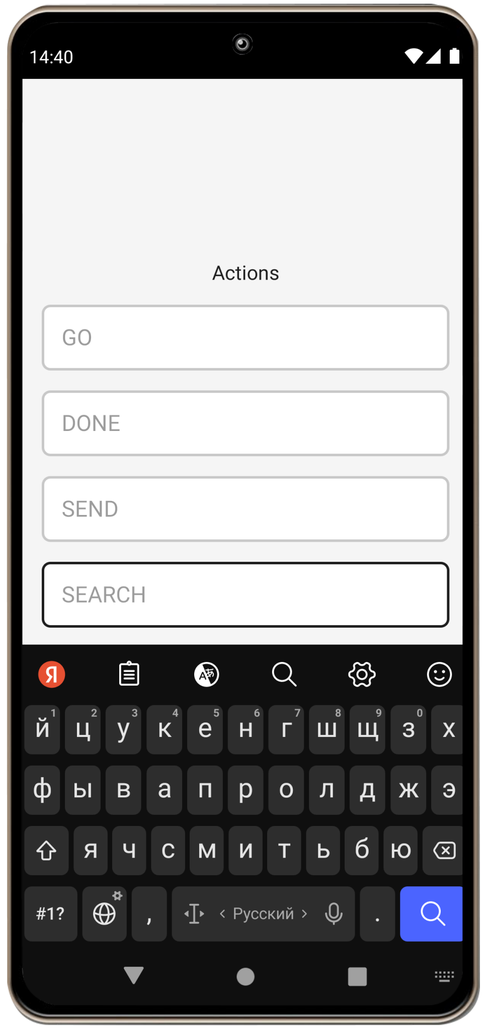




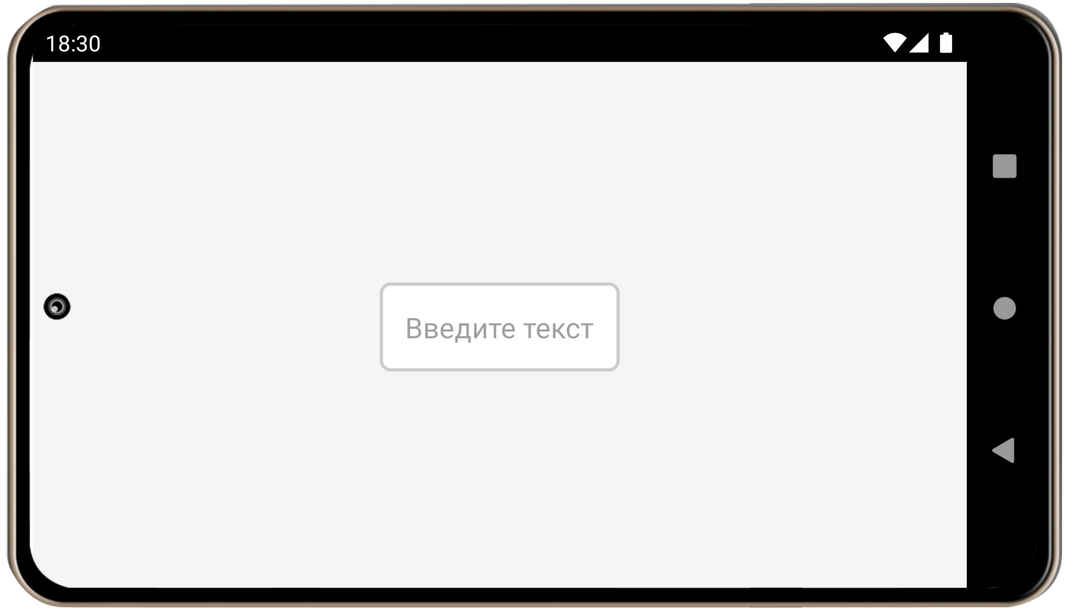
Атрибут imeOptions

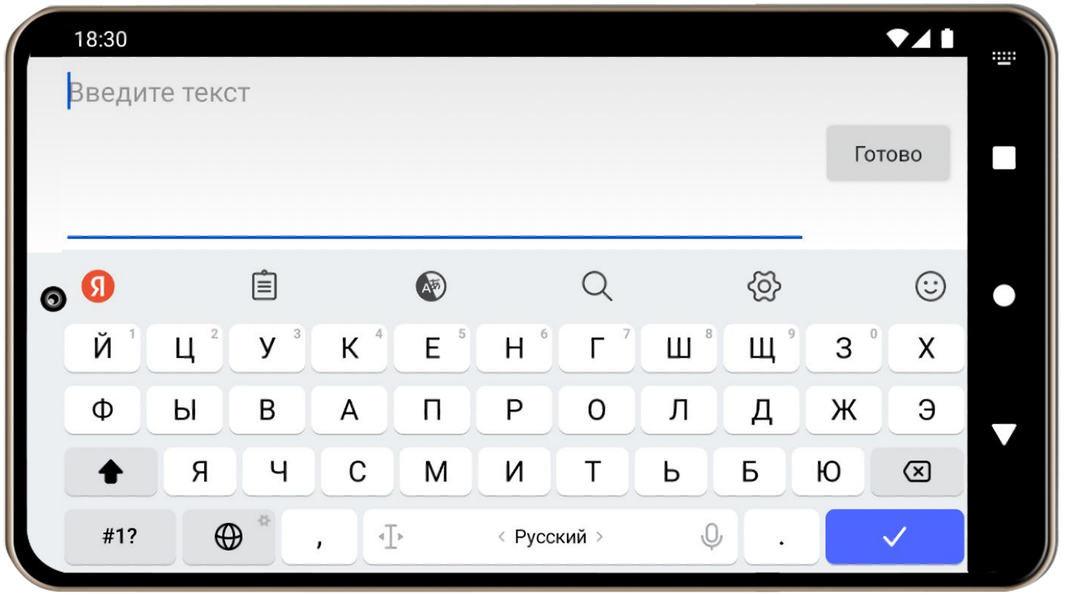






Полноэкранный режим





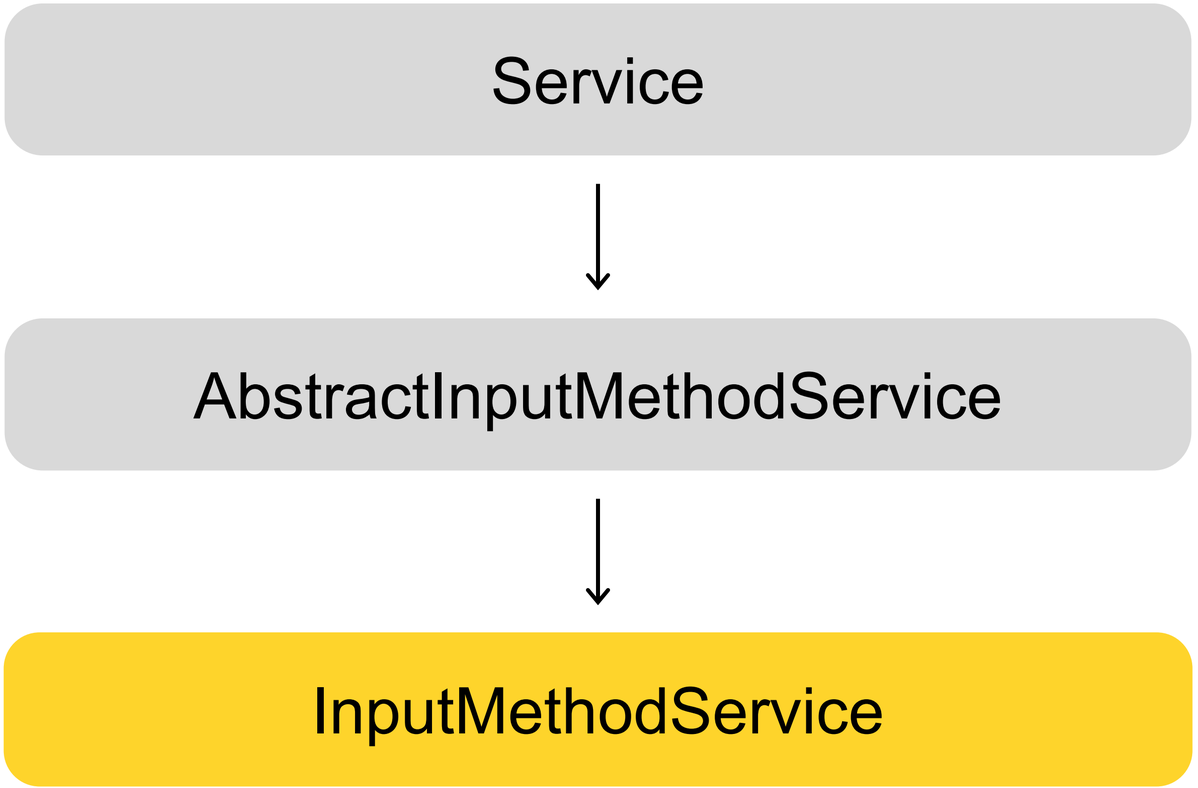
Метод onCreateInputConnection

```
override fun onCreateInputConnection(outAttrs: EditorInfo): InputConnection? {
    if (isEnabled && onCheckIsTextEditor()) {
        val editorState = editorStateFlow.value
        outAttrs.inputType = editorState.inputType
        outAttrs.imeOptions = editorState.imeOptions
        // TODO: fill in other properties
        return ExampleInputConnection()
    }
    return null
}
```

Метод setOnEditorActionListener

```
editText.setOnEditorActionListener { _, actionId, _ ->
    if (actionId == EditorInfo.IME_ACTION_SEND) {
        sendMessage()
        return true
    }
    return false
}
```

InputMethodService



Метод onCreate

- Создание служебных вьюх
- Инициализация зависимостей

Метод onStartInput

- Обработка информации из EditorInfo
- Подготовка компонентов к началу ввода

Метод onCreateInputView

- Создание клавиатурной вью
- Инициализация дополнительной вью-логики

Метод onStartInputView

- Отображение клавиатурной вью
- Инициализация основной вью-логики

Метод onComputeInsets

- Настройка области взаимодействия с клавиатурой поверх хост-приложения
- Возможность реализации «плавающего» режима или вспомогательных UI-элементов

Метод onFinishInputView

- Завершение функционирования вью-логики
- Скрытие клавиатурной вью или переключение на другое поле ввода

Метод onFinishInput

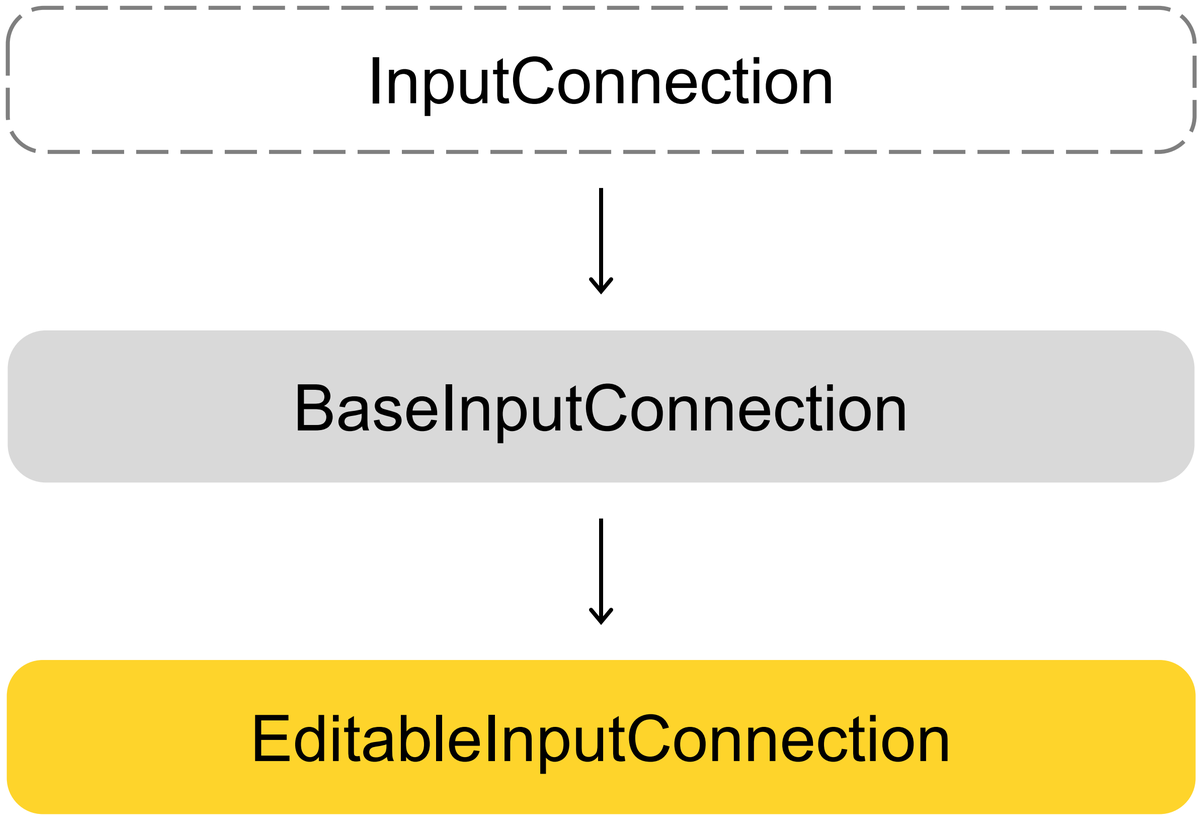
- Завершение сессии ввода
- Завершение композиции текста в реализации по умолчанию

Метод onDestroy

- Освобождение ресурсов

Поддержка физических клавиатур

InputConnection



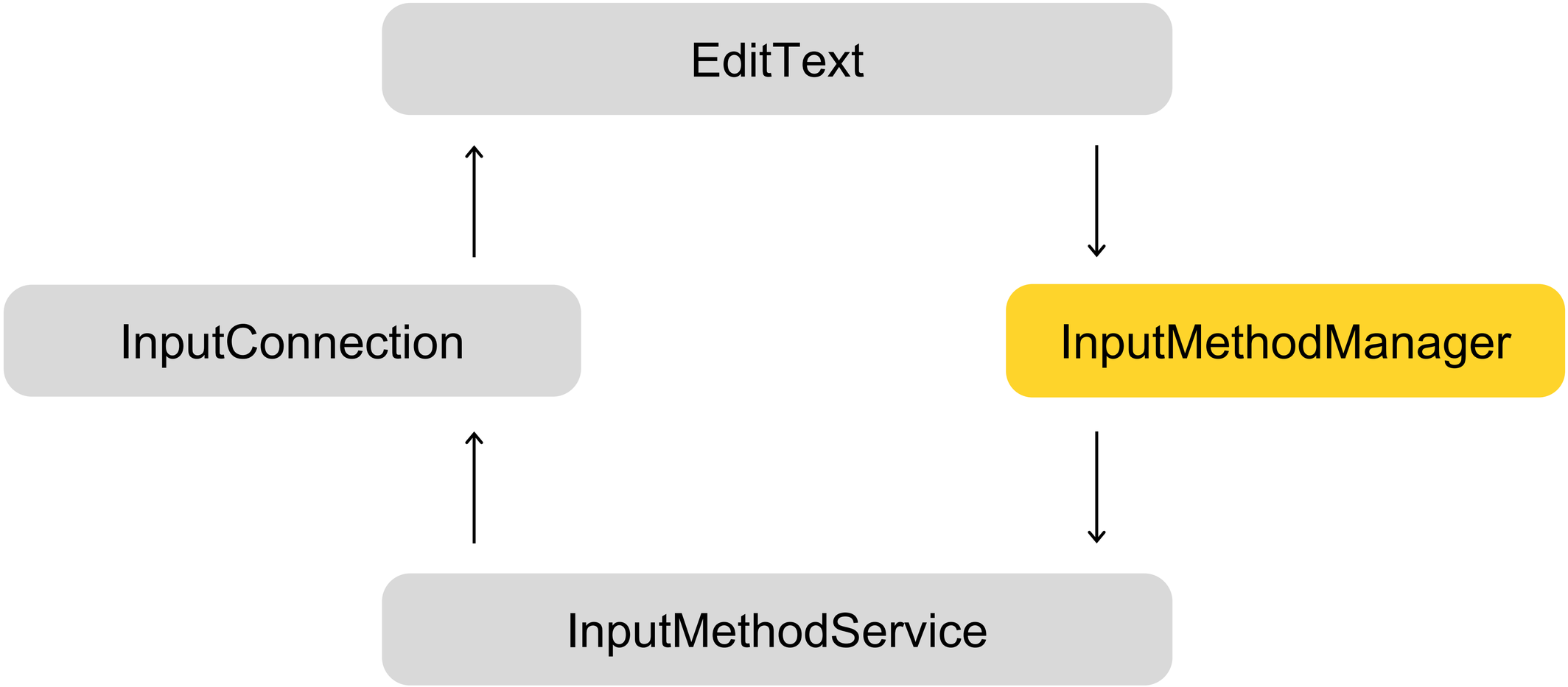
Композиция текста

```
val inputConnection = currentInputConnection

inputConnection.setComposingRegion(

    cursorPosition - numberOfCharsBeforeCursor,

    cursorPosition + numberOfCharsAfterCursor

)

// Replace currently composing text

inputConnection.setComposingText(composingText, newCursorPosition)

// Remove composing spans and finish session

inputConnection.finishComposingText()
```

Коммитинг текста и контента

```
val inputConnection = currentInputConnection

// Commit text

inputConnection.commitText(text, text.codePointCount(0, text.length))

// Commit content

InputConnectionCompat.commitContent(

    inputConnection,

    editorInfo,

    inputContentInfo, // Contains content Uri and MimeType

    commitContentFlags,

    null

)
```

InputMethodManager



Управление клавиатурой

```
val inputMethodManager =
    context.getSystemService(Context.INPUT_METHOD_SERVICE) as InputMethodManager
// Show keyboard
inputMethodManager.showSoftInput(view, 0)
// Refresh keyboard
inputMethodManager.restartInput(view)
// Hide keyboard
inputMethodManager.hideSoftInputFromWindow(view.windowToken, 0)
```

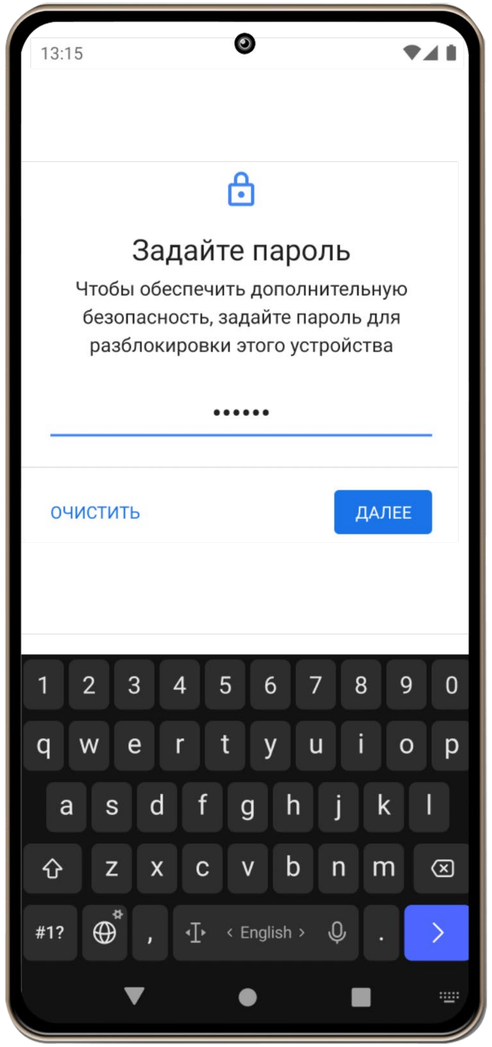
Получение сведений о методах ввода

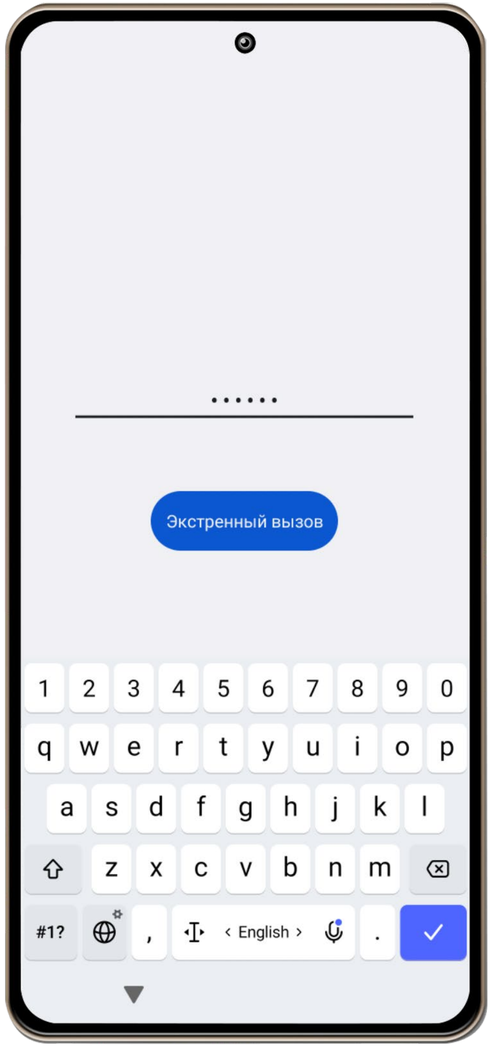
```
val inputMethodManager =
    context.getSystemService(Context.INPUT_METHOD_SERVICE) as InputMethodManager
// Get a list of all input methods
inputMethodManager.inputMethodList
// Get a list of enabled input methods
inputMethodManager.enabledInputMethodList
// Get the current input method subtype
inputMethodManager.currentInputMethodSubtype
```

04 Яндекс Клавиатура

Режим DirectBoot

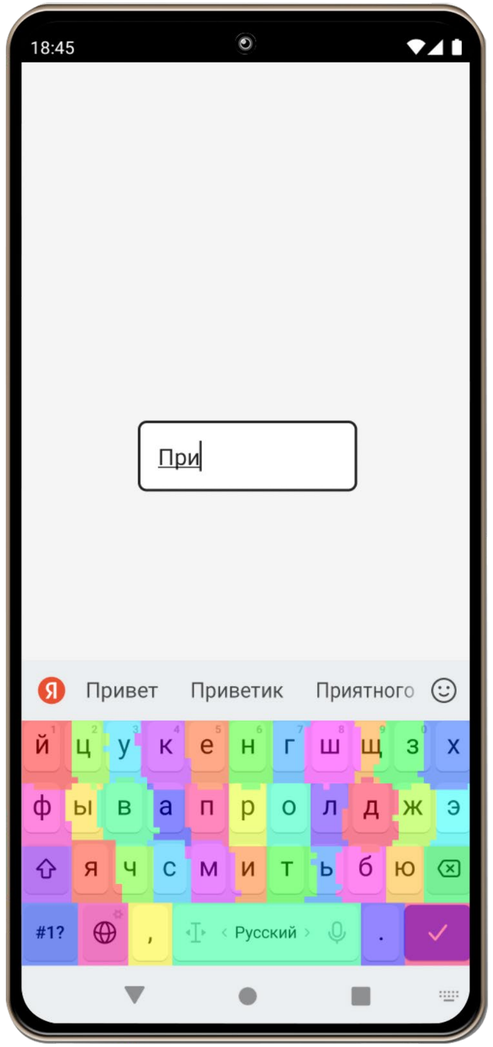
- Доступен с Android 7+
- Позволяет работать только с зашифрованным хранилищем устройства
- Настройка компонента на поддержку режима выполняется через атрибут directBootAware
- Активность режима контролируется с помощью ACTION_USER_UNLOCKED и UserManager#isUserUnlocked

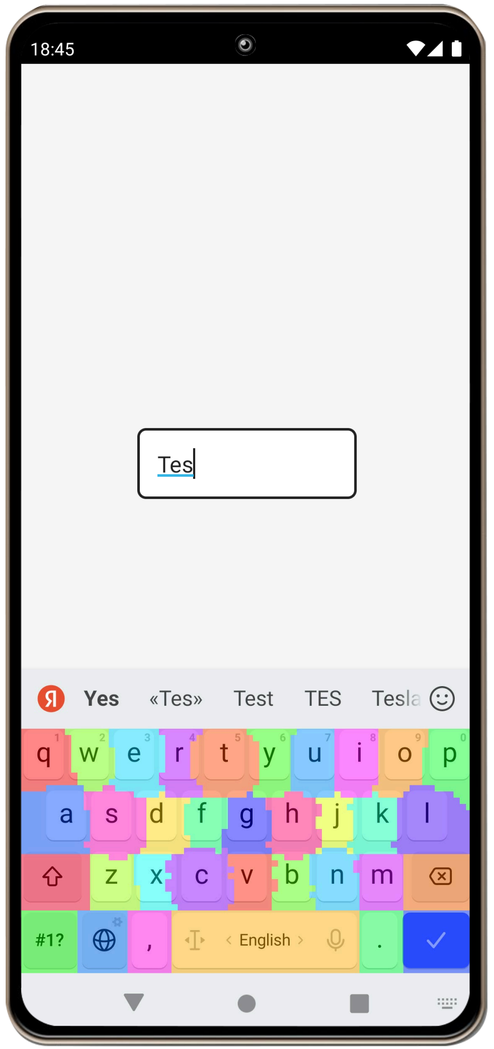




Динамическая сетка

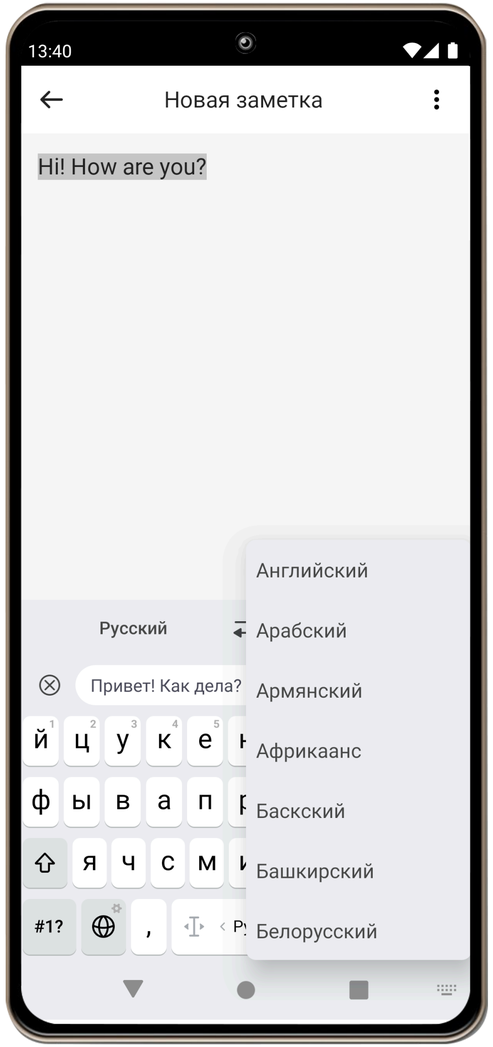
- Динамическое изменение тач-зоны на основе предиктивного алгоритма
- Уменьшение кол-ва опечаток на 15% и увеличение скорости ввода
- Адаптация ввода под разные форматы экранов





Переводчик текста

- Выполняет синхронный перевод
- Поддерживает более 100 языков
- Помогает преодолевать языковые барьеры



Яндекс

Mobius

Спасибо!

Вадим Черненко

Старший разработчик

Дмитрий Дегтярёв

Руководитель разработки