**Intent**

Intent - это фундаментальный компонент операционной системы Android, который обеспечивает связь между различными частями приложения или между разными приложениями. Он служит механизмом передачи сообщений и позволяет запрашивать определенные действия или передавать данные между компонентами.

Вот несколько распространенных вариантов использования Intent:

* запуск активностей: Чаще всего Intent’ы используются для запуска новых активностей в приложении.
* обмен данными между активностями: Функции Intent позволяют передавать данные от одной активности к другой. Это используется, когда необходимо передать информацию между различными частями приложения;
* использование сторонних действий - сервисов Android: Intent используются для вызова общих функций ОС Android таких как: открытие веб-ссылок, отправка электронных писем или телефонных звонков;
* получение информации от сторонних приложений: Intent позволяет получать информацию о результатах выполнения сервисов или подписываться на другие приложения и получать информацию от них.

Intent’ы бывают двух типов: Явные и Неявные

Явные используют имя компонента, для реализации внутреннего подключения приложения, для перемещения внутри приложения и передачи данных.

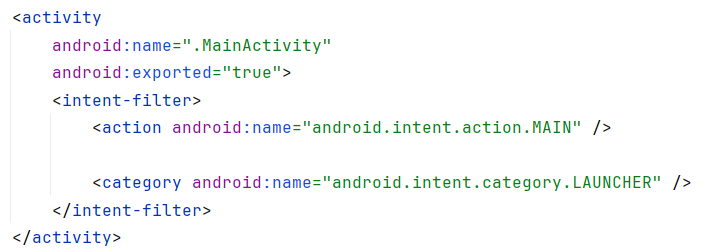
Переход между экранами:



В примере используется класс SecondActivity - это Java-класс, на активность которого будет осуществляться переход.

Для реализации перехода на SecondActivity необходимо объявить класс в файле AndroidManifest.xml.

В AndroidManifest.xml необходимо объявить классы которые используются для отображения экранов, например начальный экран по умолчанию MainActivity



Все используемые для отображения классы должны быть описаны в файле AndroidManifest.xml с помощью элемента <activity>. Каждый подобный элемент содержит как минимум один атрибут android:name, который устанавливает имя класса.

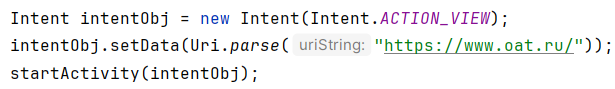
Для MainActivity в элементе intent-filter определяется интент-фильтр. В нем значение "android.intent.action.MAIN" представляет главную точку входа в приложение.

Создайте элемент для SecondActivity, указывать фильт нет необходимости.



Неявным намерениям указывают действие, которое должно быть выполнено, и в дальнейшем это действие обрабатывается компонентом другого приложения. В отличие от явного Intent, вы не используете имя класса при передаче.

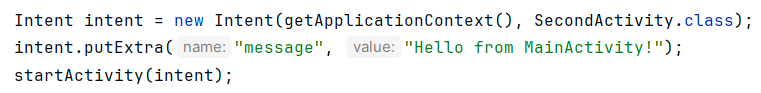
Например, реализуем переход на веб-страницу:



Передача данных с помощью Intent:

Intent позволяет передавать данные между компонентами, что необходимо, когда мы хотим отправить информацию из одного экрана на другой.

**Отправка данных:** Для отправки данных вы можете использовать Extras. Extras - это пары ключ-значение, которые могут быть добавлены к Intent для передачи дополнительной информации.



В этом примере Intent содержит строковое сообщение с ключом "message".

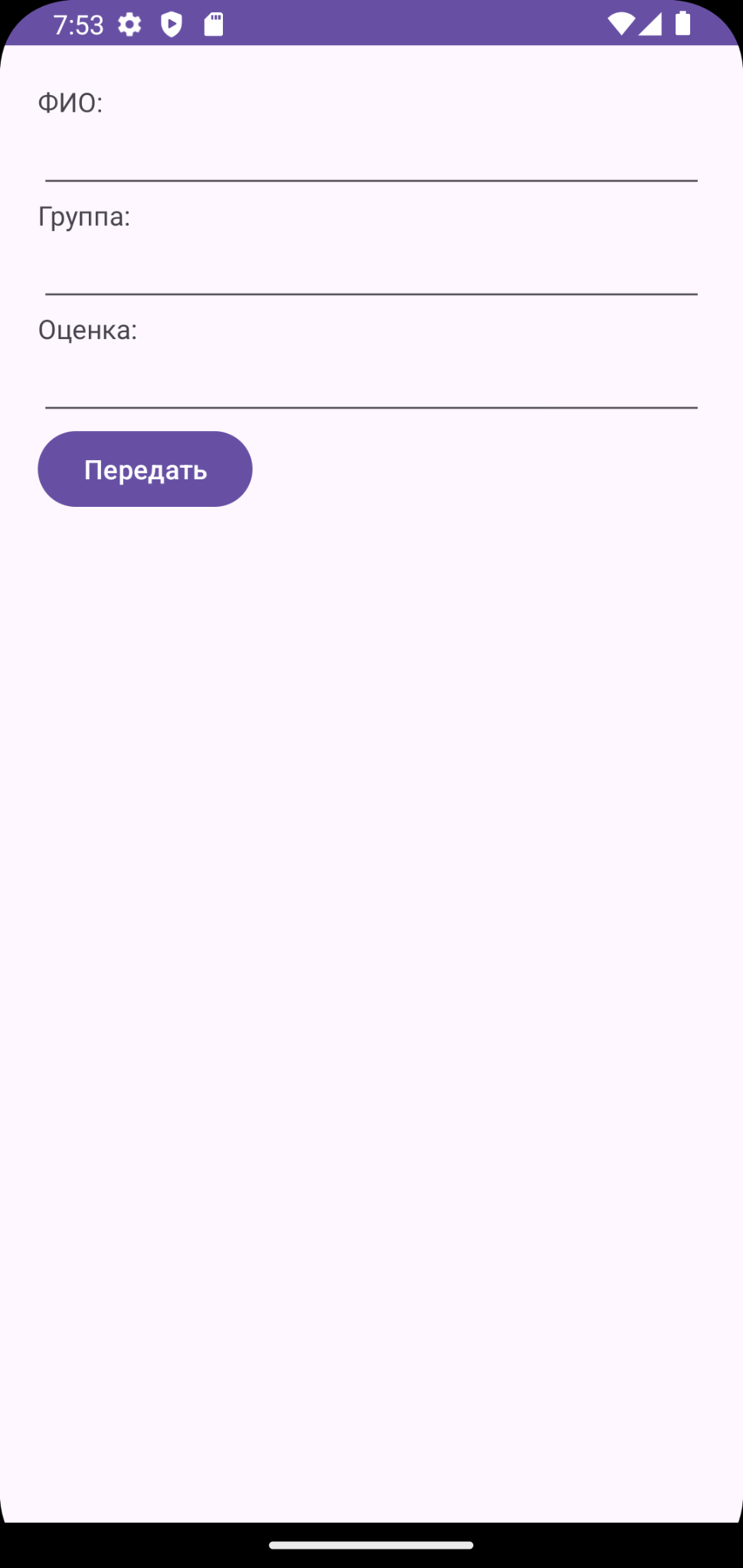
**Получение данных:** Чтобы получать данные, отправленные через Intent, принимающий компонент должен извлечь дополнительные данные.



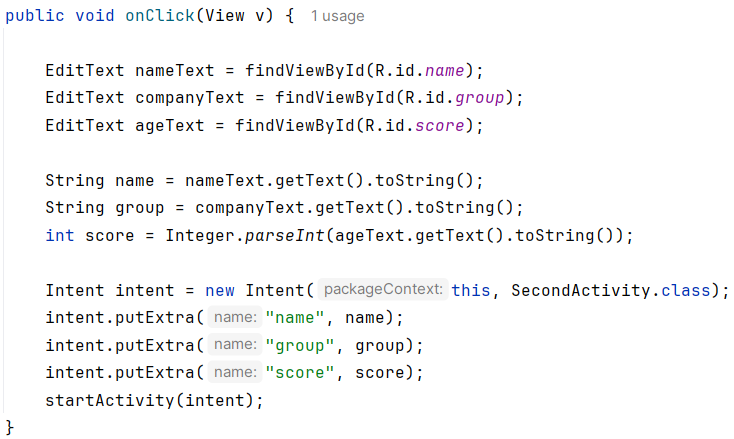
В зависимости от типа отправляемых данных при их получении можно использовать ряд методов объекта Bundle. Все они в качестве параметра принимают ключ объекта.

* get(): универсальный метод, который возвращает значение типа Object. Соответственно поле получения данное значение необходимо преобразовать к нужному типу
* getString(), getInt(), getByte(), getSerializable() и т.д.: возвращают объект соответствующего типа String

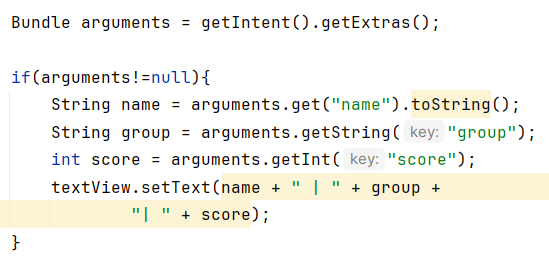
Разработайте следующий интерфейс для MainActivity:



Реализуйте в MainActivity функцию получения и передачи данных через Intent:



Реализуйте функцию в SecondActivity для получения данных из объекта Bundle. Предполагается, что данному экрану будут передаваться три элемента - две строки с ключами name и group и число с ключом score:



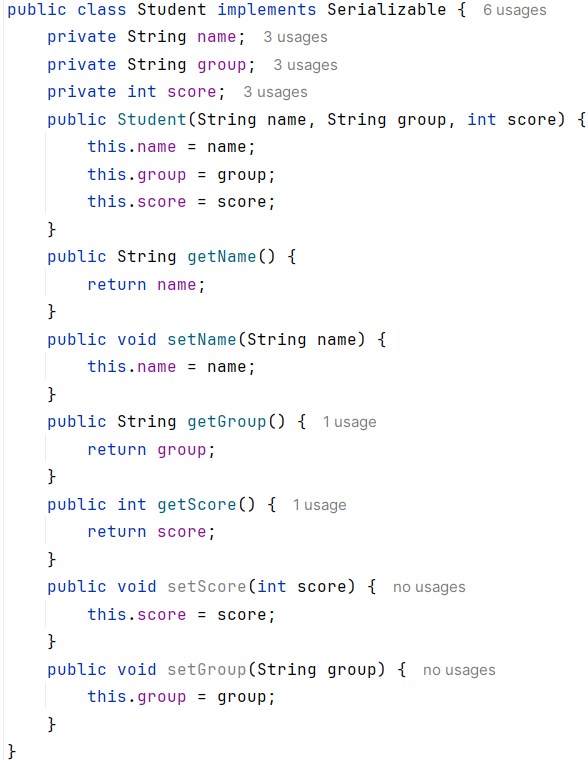
В случае успешного запуска приложения, при заполнении данных и нажатии на кнопку, откроется второй экран с установленными ранее данными:



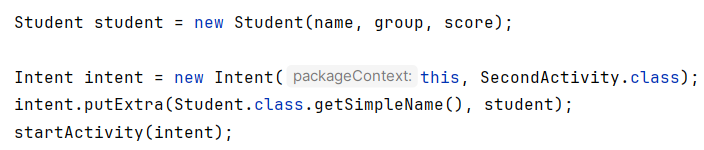
Передача сложных объектов

Intent позволяют передавать как примитивные типы данных, так и сложные объекты. Для реализации передачи сложных объектов необходимо создать класс для передаваемых данных, чтобы создать объекты этого класса, и реализовать механизм сериализации.

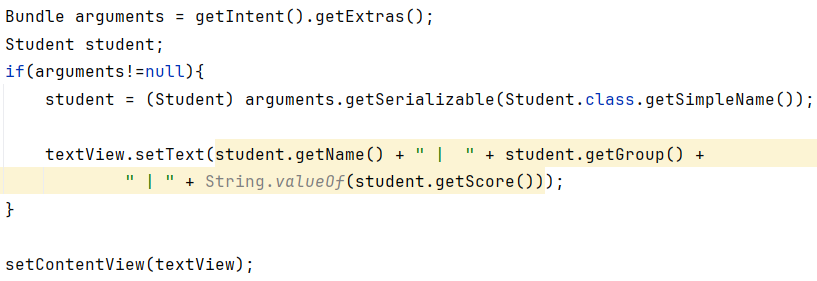
Создайте новый класс Student, реализуйте в нём необходимые переменные, конструктор, гетеры и сетеры. Класс должен реализовывать интерфейс Serializable:



Переработайте класс MainActivity, изменив передаваемый тип данных, с нескольких разрозненных переменных в единый объект класса Student. В качестве ключа используется результат метода Student.class.getSimpleName(), возвращающий название класса.



Далее, переработайте класс SecondActivity, изменив получаемый тип данных. Для получения объекта применяется метод getSerializable(), поскольку класс Student реализует интерфейс Serializable.



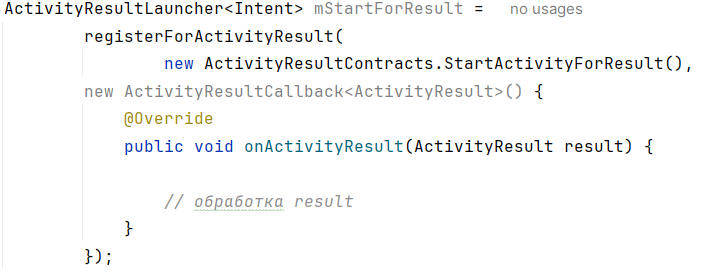
**Обработка результатов**

Intent может не только запустить новый экран но и получать из него данные. Для получения результата работы запускаемой activity необходимо использовать Activity Result API.

Activity Result API предоставляет компоненты для регистрации, запуска и обработки результата другой Activity. Одним из преимуществ применения Activity Result API является то, что он отвязывает результат Activity от самой Activity. Это позволяет получить и обработать результат, даже если Activity, которая возвращает результат, в силу ограничений памяти или в силу других причин завершила свою работу.

Регистрация функции для получения результата

Для регистрации функции, которая будет обрабатывать результат, Activity Result API предоставляет метод registerForActivityResult(). Этот метод в качестве параметров принимает объекты ActivityResultContract и ActivityResultCallback и возвращает объект ActivityResultLauncher, который применяется для запуска другого класса.



* ActivityResultContract определяет контракт: данные какого типа будут подаваться на вход и какой тип будет представлять результат.
* ActivityResultCallback представляет интерфейс с единственным методом onActivityResult(), который определяет обработку полученного результата. Когда вторая activity закончит работу и возвращает результат, то будет вызываться данный метод. Результат передается в метод в качестве параметра result.

Запуск activity для получения результата

Метод registerForActivityResult() регистрирует и возвращает объект ActivityResultLauncher. С помощью этого вы можем запустить activity. Для этого у объекта ActivityResultLauncher вызывается метод launch():



В метод lauch() передается объект того типа, который определен объектом ActivityResultContracts в качестве входного.

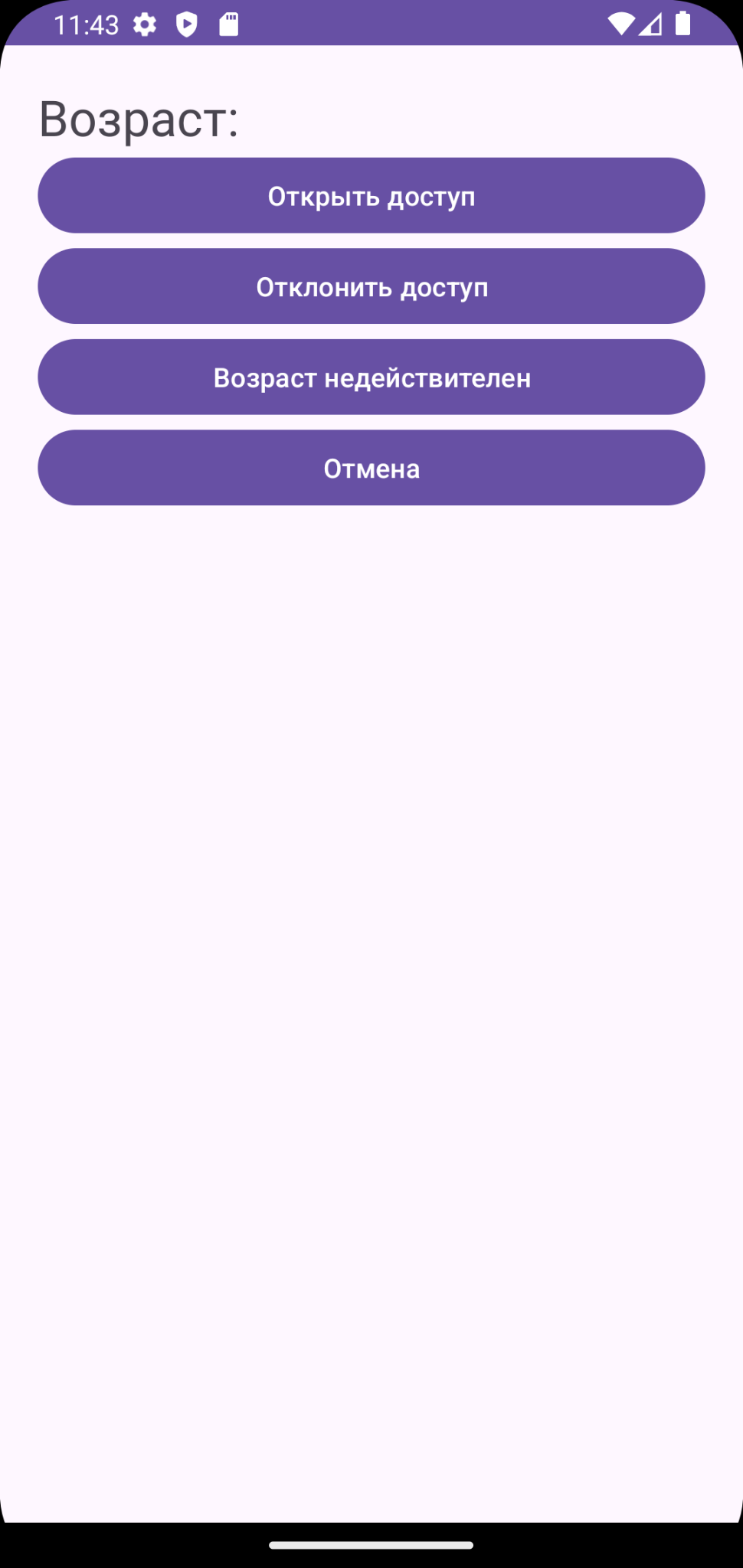
Пример обработки результатов

Создайте два класса и разработайте для них следующий макет интерфейса.

Для первого экрана:



Для второго экрана:



Реализуйте в первом классе функции передачи данных и получения ответа от экрана.

Определите объект ActivityResultLauncher, с помощью которого будем запускается второй экран и передаются данные. Объект ActivityResultLauncher типизируется типом Intent, так как объект этого типа будет передаваться в метод launch() при запуске второго экрана.

Тип контракта определяется типом ActivityResultContracts.StartActivityForResult, который определяет тип Intent в качестве входного типа и тип ActivityResult в качестве возвращаемого типа.

Второй аргумент метода registerForActivityResult() - объект ActivityResultCallback типизируется типом результата - типом ActivityResult и определяет возвращаемую функцию onActivityResult(), которая получает результат и обрабатывает его.

В качестве результата, как правило, применяются встроенные константы Activity.RESULT\_OK и Activity.RESULT\_CANCELED, которые означаю успешное подтверждение или отклонение запроса, соответвенно.

С помощью метода getData() результата получаете переданные из второго экрана данные в виде объекта Intent:

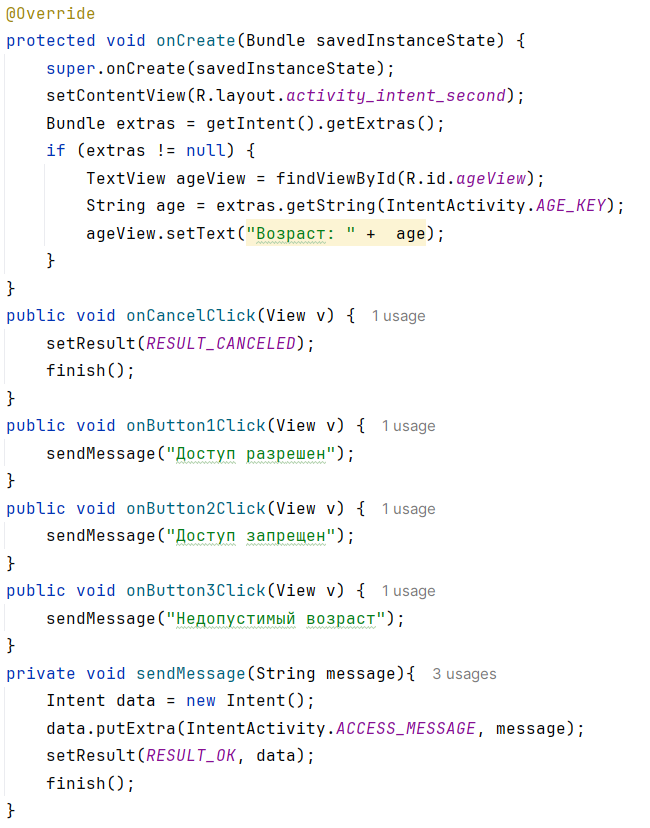


Реализуйте во втором классе функции получения данных и настройки возвращаемых данных.

Три кнопки вызывают метод sendMessage(), в который передают отправляемый ответ. Метод отвечает за сообщение которое будет возвращено первому экрану.

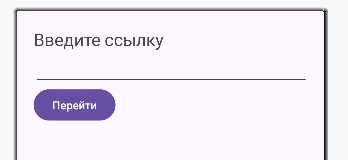
Для возврата результата необходимо вызвать метод setResult(), в который передаются два параметра:

* числовой код результата
* отправляемые данные



**Задание для самостоятельной работы:**

1. Разработайте приложение с интерфейсом соответствующим следующему макету. Реализуйте функцию которая открывает введенную в поле гиперссылку.



1. Доработайте приложение из СРС№3. Разработайте приложение с интерфейсом соответствующим следующему макету. При нажатии на кнопку открывается второй экран который имеет цвет выбранной кнопки. Добавьте кнопку возврата на первый экран.

