

Лабораторная работа 1.

«Создание формы ввода данных с валидацией»

Цель: Научиться создавать интерфейс формы ввода данных, реализовывать клиентскую валидацию и обработку ошибок.

Задание: Создать интерактивный пользовательский интерфейс с валидацией полей, обработкой ошибок и отображением Toast-уведомлений при успешной отправке.

Номер варианта соответствует последней цифре Вашего пароля.

Среда разработки: Android Studio.

Язык: Kotlin.

Вариант 1: Форма регистрации пользователя.

Порядок выполнения:

1. Создайте активность с полями:
 - Имя (только кириллица, 2+ символа)
 - Email (валидация формата)
 - Пароль (скрытый ввод, 8+ символов, буквы верхнего/нижнего регистра, цифры).
 - Чекбокс "Принимаю условия".
 - Кнопка "Зарегистрироваться".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Пустое имя: "Поле не может быть пустым".
 - Неверный email: "Некорректный email".
 - Не отмечен чекбокс: "Примите условия".
4. При успешной валидации показывать Toast "Регистрация успешна".

Вариант 2: Форма оплаты банковской картой.

Порядок выполнения:

1. Создайте активность с полями:
 - Номер карты (16 цифр с пробелами через 4 символа).
 - Срок действия (ММ/YY с автодобавлением "/").
 - CVV (3 цифры).
 - Имя владельца (только латиница).
 - Кнопка "Оплатить".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Неполный номер: "Введите 16 цифр"
 - Просроченная карта: "Срок действия истек"
 - Неверный CVV: "3 цифры на обороте"
4. При успешной валидации показывать Toast "Оплата принята".

Вариант 3: Форма бронирования отеля.

Порядок выполнения:

1. Создайте активность с полями:
 - Дата заезда (дд/мм/гггг).
 - Дата выезда (минимум на 1 день > заезда).
 - Количество гостей (1-6).
 - Тип номера (список: Эконом/Стандарт/Люкс).
 - Кнопка "Забронировать".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Неверный формат даты заезда.
 - Обратная дата: "Дата выезда позже заезда".
 - Превышение гостей: "Максимум 6 гостей".
4. При успешной валидации показывать Toast "Бронь подтверждена".

Вариант 4: Форма заказа доставки.

Порядок выполнения:

1. Создайте активность с полями:
 - Адрес (город, улица, дом, квартира)
 - Время доставки (выбор из интервалов)
 - Телефон (формат +7 XXX XXX-XX-XX)
 - Комментарий курьеру.
 - Кнопка "Заказать".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Неполный адрес: "Уточните адрес"
 - Неверный телефон: "Формат +7 XXX XXX-XX-XX"
 - Укажите комментарий, например, код домофона или номер подъезда.
4. При успешной валидации показывать Toast "Заказ создан".

Вариант 5: Форма обратной связи.

Порядок выполнения:

1. Создайте активность с полями:
 - Тема (список: Ошибка/Предложение/Вопрос).
 - Сообщение (мах. 150 символов).
 - Email (валидация формата).
 - Дополнительная информация.
 - Кнопка "Отправить обращение".

2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Поле «сообщение» не должно быть пустым.
 - Неверный email: "Некорректный email".
 - Укажите дополнительную информацию, которая может быть полезна.
4. При успешной валидации показывать Toast "Обращение отправлено".

Вариант 6: Форма создания события.

Порядок выполнения:

1. Создайте активность с полями:
 - Название (мин. 5 символов)
 - Дата/время (будущее время, дд/мм/гггг)
 - Продолжительность (15-360 минут)
 - Описание (макс. 1000 символов)
 - Кнопка "Создать".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Прошедшая дата: "Выберите будущее время"
 - Короткое название: "Минимум 5 символов"
 - Неверная длительность: "15-360 минут"
4. При успешной валидации показывать Toast "Событие создано".

Вариант 7: Форма поиска недвижимости.

Порядок выполнения:

1. Создайте активность с полями:
 - Цена максимальная, руб.
 - Количество комнат (1-4).
 - Район (выпадающий список).
 - Ипотека, наличные (чекбокс).
 - Кнопка "Найти".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Цена не указана.
 - Количество комнат не может быть меньше 1.
 - Не отмечен чекбокс о расчете.
4. При успешной валидации показывать Toast "Поиск начат".

Вариант 8: Форма медицинской анкеты.

Порядок выполнения:

1. Создайте активность с полями:
 - Рост, см (количество цифр: 2-3).

- Вес, кг.
 - Хронические заболевания (выбор из списка)
 - Аллергии (чекбокс)
 - Кнопка "Сохранить".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
 3. При невалидных данных показывайте ошибки:
 - Поле не может быть пустым.
 - Укажите корректный вес не менее 1 кг.
 - Выберите заболевание или его отсутствие.
 4. При успешной валидации показывать Toast "Анкета сохранена".

Вариант 9: Форма подписки на услугу.

Порядок выполнения:

1. Создайте активность с полями:
 - Тариф (радиокнопки: Базовый/Премиум).
 - Период оплаты (список: Месяц/Год).
 - Промокод (с проверкой на формат: буква, три цифры).
 - Согласие на списание (чекбокс).
 - Кнопка "Оформить".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Укажите период оплаты.
 - Неверный промокод.
 - Укажите согласие на списание.
4. При успешной валидации показывать Toast "Подписка оформлена".

Вариант 0: Форма заявки на курсы.

Порядок выполнения:

1. Создайте активность с полями:
 - Имя (только кириллица, 2+ символа).
 - Название курсов (список тем).
 - Телефон (формат +7 XXX XXX-XX-XX)
 - Согласие на обработку данных (чекбокс).
 - Кнопка "Оставить заявку".
2. Используйте TextInputLayout для отображения ошибок валидации формы.
3. При невалидных данных показывайте ошибки:
 - Пустое имя: "Поле не может быть пустым".
 - Неверный телефон: "Формат +7 XXX XXX-XX-XX"
 - Укажите согласие на обработку данных.
4. При успешной валидации показывать Toast "Заявка отправлена".

Методические указания к выполнению лабораторной работы №1

Для выполнения лабораторной работы №1 необходимо изучить лекции 1-3.
Затем установить и настроить среду разработки:

1. Установка Android Studio.

- Скачайте Android Studio с официального сайта <https://developer.android.com/>.
- Установите программу, следуя инструкциям установщика.
- При установке активируйте галочки:
 - ✓ Android SDK
 - ✓ Android Virtual Device

2. Настройка SDK и эмулятора :

- Откройте Android Studio → Configure → SDK Manager .
- Убедитесь, что установлены:
 - ✓ Android SDK (версия 21 и выше)
 - ✓ Android Emulator
 - ✓ Android SDK Build-Tools
- В разделе System Settings включите автоматическую загрузку обновлений.
- Создайте виртуальное устройство (AVD) через Device Manager для тестирования.

3. Проверка настройки Kotlin :

- При создании нового проекта выберите язык Kotlin .
- Убедитесь, что в build.gradle.kts (уровень проекта) присутствует плагин Kotlin:

```
plugins {  
    id("com.android.application") version "8.0.2" apply false  
    id("org.jetbrains.kotlin.android") version "1.8.0" apply false  
}
```

После успешной установки создайте новый проект:

1. Откройте Android Studio → New Project → Empty Activity .

2. Укажите:

- Name : например, FormValidationApp
- Package name : com.example.formvalidation
- Language : Kotlin

3. Нажмите Finish.

После выполнения данных шагов в папке res/layout будет файл activity_main.xml (интерфейс), а в папке java/.../MainActivity.kt — основная активность.

Далее необходимо добавить библиотеки Material Design в проект. Для этого выполните следующие шаги:

- Шаг 1: Открытие файла build.gradle.kts (уровень модуля)

В Android-проектах на Kotlin используется система сборки Gradle, которая управляет зависимостями (библиотеками) и настройками проекта. У каждого

проекта есть несколько файлов `build.gradle.kts`. Нас интересует файл уровня модуля, обычно называется `build.gradle.kts` внутри папки `app`. Чтобы его найти, перейдите в панель Project (если её нет — нажмите View → Tool Windows → Project). Затем найдите файл `build.gradle.kts` и откройте его двойным кликом.

Этот файл содержит информацию о том, какие библиотеки используются в вашем приложении, какая версия Android поддерживается, и как строится сам проект.

- Шаг 2: Добавление зависимости Material Components

Зависимость — это ссылка на внешнюю библиотеку, которую можно подключить к проекту, чтобы использовать её функционал. Библиотека Material Components находится в репозитории Google. Добавьте следующую строку в секцию `dependencies` вашего `build.gradle.kts`:

```
implementation("com.google.android.material:material:1.9.0")
```

Здесь `implementation` — это тип зависимости, указывающий, что эта библиотека будет использоваться только в этом модуле (не будет доступна другим модулям).

"com.google.android.material:material:1.9.0" — идентификатор библиотеки:

`com.google.android.material` — группа разработчика; `material` — имя библиотеки;

`1.9.0` — версия библиотеки. Рекомендуется использовать актуальную стабильную версию, так как в новых версиях исправлены ошибки и добавлены улучшения (проверьте актуальность версии на текущий момент:

<https://central.sonatype.com/search?q=Material%20Components>).

- Шаг 3: Синхронизация проекта

После того как вы добавили новую зависимость, Gradle должен загрузить эту библиотеку из интернета и интегрировать её в ваш проект. Для этого необходимо выполнить синхронизацию. После сохранения файла `build.gradle.kts` Android Studio покажет всплывающее окно в правом верхнем углу с надписью `Sync Now`. Нажмите на него, чтобы запустить синхронизацию. Если окно не появилось, то в меню сверху выберите File → Sync Project with Gradle Files.

Дождитесь завершения синхронизации — внизу Android Studio появится сообщение `Gradle sync finished`.

После добавления зависимости и синхронизации проекта вы получаете доступ ко всем компонентам, стилям и возможностям, которые предлагает Material Design.

Теперь можно перейти к проектированию проекта. Используйте XML для создания формы:

- Откройте `res/layout/activity_main.xml` и переключитесь на вкладку Design.

Добавьте элементы для каждого поля формы.

- Используйте `ConstraintLayout` или `LinearLayout` для упорядочивания элементов.

- Для хранения текстовых сообщений (например, ошибок валидации) рекомендуется использовать файл `res/values/strings.xml`. Например:

```
<!-- res/values/strings.xml -->
```

```
<resources>
```

```
<string name="error_empty_name">Поле не может быть пустым</string>
```

```
<string name="error_invalid_email">Некорректный email</string>
</resources>
```

Затем обращайтесь к строкам через ресурсы. Например:

```
val errorMessage = getString(R.string.error_empty_name)
textInputLayout.error = errorMessage
```

- В MainActivity.kt определите методы проверки для каждого поля.

Например, проверка длины строки:

```
fun validateLength(input: String, minLength: Int): Boolean {
    return input.length >= minLength }

```

- Для каждого поля используйте setError и setErrorEnabled. Например:

```
val nameInputLayout =
    findViewById<TextInputLayout>(R.id.nameInputLayout)
val nameEditText = findViewById<TextInputEditText>(R.id.nameEditText)
if (nameEditText.text.isNullOrEmpty()) {
    nameInputLayout.error = "Поле не может быть пустым"
    return false
} else {
    nameInputLayout.error = null
}

```

- Добавьте обработчик нажатия на кнопку. Например:

```
val submitButton = findViewById<Button>(R.id.submitButton)
submitButton.setOnClickListener {
    if (validateForm()) {
        Toast.makeText(this, "Форма успешно отправлена",
            Toast.LENGTH_SHORT).show()
    }
}

```

- Для запуска приложения на эмуляторе выберите виртуальное устройство из меню Run .
- Протестируйте все сценарии: корректный ввод данных, ошибки валидации, отправка формы.

Требования к отчету

Результаты выполненной лабораторной работы должны быть оформлены в формате текстового редактора Word, размером шрифта 14 пунктов и включать:

1. Титульный лист.
2. Текст задания, соответствующий Вашему варианту.
3. Описание структуры проекта, алгоритма выполнения, используемых функций.
4. Результаты работы программы в виде скриншотов работающего приложения.
5. Программный код с комментариями.
6. Ссылки на источники внешней информации, которые были использованы при выполнении заданий.

Объем пояснительной записки должен включать в себя не менее 5 страниц поясняющего текста (не считая исходного кода программ).

Список дополнительных источников

1. Гриффитс Дон, Дэвид. Программирование для Android на Kotlin, 3-е изд. «Питер», Серия «Head First O'Reilly», 2023. – 912 с.
(<https://disk.yandex.ru/i/qNiAahI4WTtCnQ>)
2. Жемеров Д., Исакова С. Kotlin в действии, ДМК Пресс, 2017. – 404 с.
3. Скин Д., Гринхол Д., Бэйли Э. Kotlin. Программирование для профессионалов, 2-е изд., Питер, 2023 – 560 с.