28 ViewModel в Dessert Clicker

[1. Прежде чем начать](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#0)

Введение

Dessert Clicker работает со встроенным состоянием и данными. В этом упражнении вы удалите встроенное состояние, данные и логику из MainActivity и переместите их в файл ViewModel.

Абстрагирование логики приложения из представления в ViewModel — это современная практика разработки под Android. Эта практика дает следующие преимущества:

* Код становится более читабельным для других разработчиков.
* Код становится более тестируемым.
* Несколько разработчиков могут одновременно работать над приложением, не мешая работе других разработчиков.

Код решения доступен в конце, но попробуйте решить упражнения, прежде чем проверять ответы. Рассматривайте решение как один из способов реализации приложения.

Предварительные условия

* Курсовая работа «Основы Android в Compose» с помощью лаборатории кода [ViewModel и State in Compose.](https://developer.android.com/codelabs/basic-android-kotlin-compose-viewmodel-and-state" \t "_blank)

Что вам понадобится

* Компьютер с доступом в Интернет и установленной Android Studio.
* Код решения для приложения Dessert Clicker

Что ты построишь

В этих практических задачах вы улучшите архитектуру приложения Dessert Clicker, добавив объект ViewModelдля обработки данных и логики приложения.

Практические задачи разделены на разделы, в которых вы индивидуально выполните следующие шаги:

* Обновите и добавьте необходимые зависимости.
* Создайте ViewModelкласс.

[2. Загрузите стартовый код](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#1)

**URL стартового кода:**

[https://github.com/google-developer-training/basic-android-kotlin-compose-training-dessert-clicker](https://github.com/google-developer-training/basic-android-kotlin-compose-training-dessert-clicker/tree/main)

**Название филиала со стартовым кодом:**main

1. В Android Studio откройте basic-android-kotlin-compose-training-dessert-clickerпапку.
2. Откройте код приложения Dessert Clicker в Android Studio.

[3. Настройте зависимости](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#2)

1. Добавьте следующую переменную в build.gradleфайл проекта:

buildscript {  
   ext {  
       ...  
       lifecycle\_version = '2.5.1'  
   }  
}

1. Добавьте в файл следующую зависимость app/build.gradle:

dependencies {  
    ...implementation "androidx.lifecycle:lifecycle-viewmodel-compose:$lifecycle\_version"  
}

## [4. Создайте класс состояния пользовательского интерфейса.](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#3)

В настоящее время DessertClickerApp()составной элемент MainActivityсодержит данные и состояние, которые управляют пользовательским интерфейсом.

Создайте класс данных, содержащий все необходимые данные для пользовательского интерфейса. Данные внутри этого класса заменяют данные, которыми DessertClickerApp()в данный момент управляет составной объект.

## [5. Создайте модель представления](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#4)

Создайте ViewModelкласс, используя компонент Jetpack ViewModel. Вы используете ViewModel для управления состоянием пользовательского интерфейса.

## [6. Переместите логику и данные приложения во ViewModel.](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#5)

Переместите логику из MainActivityViewModel и сделайте данные состояния пользовательского интерфейса доступными с помощью созданного вами класса состояния пользовательского интерфейса. Удалите все данные и логику управления состоянием из MainActivity.

Попробуйте подойти к этой задаче самостоятельно. При необходимости обратитесь к лаборатории кода [ViewModel и State в Compose](https://developer.android.com/codelabs/basic-android-kotlin-compose-viewmodel-and-state" \t "_blank) для получения инструкций.

## [7. Вызовите ViewModel](https://developer.android.com/codelabs/basic-android-kotlin-compose-practice-viewmodel?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-4-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-practice-viewmodel#6)

Используйте данные и методы, предоставляемые ViewModel, для управления пользовательским интерфейсом в MainActivity.