MOBIL PROGRAMLAMA

Kadir Kıvanç

1306130027

ÖDEV 1 – ToDoS

1. **ToDo Modeli**

Veri tabanında tutulacak ToDo verisinin class olarak modellenmiş hali şu şekildedir. Her bir değişken private olarak tanımlandıktan sonra setter ve getter’ları generate edilmiştir.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

1. **Database Yapısı**

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaDatabaseHandler sınıfı, ToDo verilerinin tutulacağı database’i oluşturmak için kullanılacak olan sınıftır.

db.execSQL metoduna, yukarıda oluşturulan tablo oluşturucu komut sabiti verilerek, veritabanı oluşturulurken tablo da oluşturulur.

Veritabanında Tablo’yu oluşturacak komut string’i; yukarıda tanımlanan, veritabanında tutulacak sütun adları ve veri tipleri ile birleştirerek sabit olarak tanımlandı. COLUMN\_ID sabiti, birincil anahtar ve id’leri otomatik olarak arttıracak şekilde tanımladı.

Başka bir class’tan bu constructor çağırıldığında süper class’ın constructor’una veri tabanı adı, versiyonu ve context gibi bilgiler gönderilir.

Tablo adı ve ToDo modelinin veritabanında tutulacak sütun adları sabit olarak tanımlandı.

Veritabanı adı ve versiyonu sabit olarak tanımlandı.

Veritabanı versiyonu güncelleme durumunda yapılacaklar yazılır.

1. **Veritabanında Ekle-Sil-Güncelle-Listele İşlemleri**

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaBu işlemler ToDoDAOImpl sınıfında yapılmıştır. Bu sınıfta tanımlanacak ekle, sil, güncelle gibi işlemler, bu işlemlerin net olarak görülebilmesi için ToDoDAO arayüzünde belirtildikten sonra ToDoDAOImpl sınıfına implement edilerek Override edilmiştir.

Otomatik olarak oluşturulan id geri çağırılarak, todo objesinin id’si olarak set edilir.

addTodo metodu parametre olarak Todo objesi alır, ardından bu objeyi ayrıştırdıktan sonra uygun sütunlarla eşleştiren ContentValues sınıfından üretilen values objesinde tutar. Eklenecek kayıtın id’si otomatik olarak generate edileceği için. İd ile ilgili bir şey yazılmamıştır.

**ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama**

updateTodo metodu parametre olarak Todo objesi alır, ardından bu objeyi ayrıştırdıktan sonra uygun sütunlarla eşleştiren ContentValues sınıfından üretilen values objesinde tutar.

İnput olarak alınan todo objesinin id’si ile aynı id’ye sahip olan kaydı values objesinde tutulan todo objesinin bilgileriyle güncllerle günceller.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

İnput olarak alınan todo objesinin id’si ile aynı id’ye sahip olan kaydı veritabanından siler.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

fetchAllTodos metodu veritabanından tüm todo kayıtlarını çeker ve bunu Cursor sınıfından oluşturulan cursor değişkenine atar. cursor’ın tuttuğu kayıtların sayısı 0’dan büyük ise, her bir kaydı todo objesine, her todo objesini de todos ArrayList’ine ekleyerek geriye todos ArrayList’ini return eder.

1. **todo.xml**

ekran görüntüsü içeren bir resim

Yüksek güvenilirlikle oluşturulmuş açıklamaekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaTodo objelerini ekranda liste olarak göstermek için öncelikle listenin bir tane ögesinin tasarımını yapmak gerekir. Bu tasarım CardView Layout’u kullanılarak yapılmıştır.

İç içe LinearLayoutlar içinde 6 TextView (+1 TextView Virgül) ve 2 ImageButton kullanılmıştır. ImageButtonlar’a görsel icon kaynağı verilmesi için app:srcCompat=”…” özelliği kullanılmıştır. Bu iconlar flaticon.com sitesinden svg (scalable vectoral graphic) yani vektör formatında indirildikten sonra projeye eklenmiştir.

Bu tasarımın java tarafı RecyclerView.ViewHolder classından extend edilen MyHolder class’ında yapılacaktır.

1. **list.xml**

Bu layout dosyasına sadece todo görsel yapısının yerleştirileceği RecyclerView Layout’u yerleştirilmiştir. RecyclerView ‘ın ListView’den farkı optimizasyonları kendisinin yapmasıdır.

1. **empty.xml**

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaEğer veri tabanından çekilen veri sayısı 0 ise ekrana yansıtılacak layout’tur. Bu şekilde tasarlanmıştır. empty.xml(land) dosyası da ekran yan döndürüldüğünde oluşacak tasarımı tutar.

1. **activity\_main.xml**

**ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama**

activity\_main.xml’de 1 tane floating button bulunmakta. Bu butona basılınca bir dialog açılarak kullanıcıdan yeni todo bilgilerinin alınması sağlanacaktır.

Ayrıca yukarıda gösterilen layout’lar bu xml dosyasına include edilmiştir.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

1. **fragment\_dialog.xml**

Hem todo editlemek hem de yeni todo eklemek için kullanılacaktır. Popup içine yerleştirilecektir. Yapılan işlem tipine göre açılan popup başlığı değişecektir. Eğer editleme işlemi yapılıyorsa içindeki bilgiler ilgili todo’nun bilgileriyle otomatik doldurulacaktır.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

Date Picker popup olarak açılır. Seçilen tarih solundaki date EditText’ine yazılır

Time Picker popup olarak açılır. Seçilen saat solundaki time EditText’ine yazılır

1. **MyHolder sınıfı**

**ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama**Bu sınıf todo.xml layout’unun java tarafındaki karşılığıdır. MyAdapter sınıfının inner class’ı olarak yazılmıştır.

Delete butonuna basıldığında silinmek istenen todo’nun id bilgisi, todo görsel ögesinin üzerinde taşıdığı id değişkeninden çekilir. Geriye kalan bilgiler (opsiyonel – sadece id bilgisi yeterli olur.) todo görsel ögesinin textView’lerinden alınarak yeni bir todo objesi oluşturulur.

Veritabanı işlemleri UI thread’de yapılmamalıdır. Dolayısıyla yapılmak istenen işlem türü ve todo objesi MainActivity’deki MyAsyncTask’e pass edilmiştir. Bu bilgileri alan asynctask işlem türüne göre veritabanı işlemini gerçekleştirdikten sonra listeyi yenileyecektir.

Edit butonuna basıldığında kullanıcının karşısına bir popup çıkmalı ve bu popup’da kullanıcının editlemek istediği todo bilgileri otomatik olarak gerekli yerlere yazılmalıdır. Açılan popup’ın içeriği bir fragment olacaktır. Fragmentın görsel ögeleri (örneğin editText) başka bir class’tan doğrudan setlenemez. Bu bilgileri kendi sınıfındaki değişkenlerden almalıdır. Dolayısıyla bu değişkenler setlenir. (TitleStr gibi)

todo.xml ögesinin edit ve delete butonlarının clickListener’ları setlenir.

MyAdapter’da, todo.xml ögesindeki görsel ögelerin yanı sıra ekrana yansıtılmayan ve todo.xml ögesinin üzerinde taşıdığı todo bilgilerinin hangi tooya ait olduğunun anlaşılmasını sağlayacak bir id değişkeni tutulacaktır.

todo.xml layout’undaki view’ler id’leri sayesinde burada objelere dönüştürülür. Burada kullanılan v objesi üst sınıftan gelir. Üst sınıfta, yani MyAdapter sınıfında OnCreateViewHolder metodunda todo.xml dosyası inflate edilerek v objesinde saklanır ve MyHolder’ın constructur’ı çağırılarak parametre olarak verilir.

1. **MyAdapter sınıfı**

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaBu sınıf, oluşturulan holder’ın neresine hangi bilgilerin yazılacağını belirler.

Listenin eleman sayısını buradan çeker.

ViewHolder’ın neresine hangi bilginin yazılacağını belirler. Listenin eleman sayısına göre döngü olarak çalışır.

Yukarıda anlatılan ViewHolder’da kullanılan v objesine todo.xml layoutunu inflate eder.

1. **MyAsyncTask**

Bu sınıf veritabanı işlemlerinin arka planda yapılması gerektiğinden dolayı, veri tabanı işlemlerini çağıran sınıftır. Activity elemanlarına doğrudan erişimi vardır ve eleman sayısına göre ekrana empty.xml veya recycler view’e elemanları ekleyerek list.xml layout’unu ekrana getirir. MainActivity Class’ının inner class’ıdır.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

Listeleme işlemi için herhangi bir todo objesine gerek duymaz.

Ekle-Sil-Güncelle işlemleri için bir todo’ya gerek vardır.

İşlem türüne göre

listele,

ekle-listele,

sil-listele,

güncelle-listele

işlemlerini yapar. Burada veriler sadece arrayliste taşınır ekrana verilmez.

Arka plan işlemi başlamadan önce ekrana bir proggress bar içeren layout gösterir.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

Bilgiler arrayliste çekildikten sonra arraylistin eleman sayısı 0 ise ekrana empty.xml görüntüsünü verir.

Eleman sayısı 0 dan büyükse önceki sayfalarda yapısı anlatılan MyAdapter’e arraylist’i verir. Ardından bu adapter’ı listeye basar.

1. MainActivity Class

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

+ butonuna basıldığında yeni todo eklenmesi için boş bir popup açar.

Ekran ilk kez açıldığında async task ile veri tabanından listeyi çeker.

Activity içinde fragment kullanımını sağlar.

1. **AddOrEditTodoDialogFragment Class**

Bu class fragment\_dialog.xml layout’unun java tarafındaki karşılığıdır. İçine aldığı bilgileri, popup başlığına göre ya veritabanına todo eklemek için asynctask’i çağırır ya da todo’yu güncellemek için.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

Eğer editleme işlemi yapılıyorsa açılırken todo bilgilerini otomatik olarak popuptaki ilgili yerlere yerleştirir.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklamaekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama

Todo güncelleniyorsa asynctaske işlem bilgilerini ve todo objesini gönderir.

Hangi işlemin yapıldığını kendi başlığından anlar. (dialogTitle değişkeni) Yeni todo ekleniyorsa asynctaske işlem bilgilerini ve todo objesini gönderir.

Bir tarih seçici popup’ı açar ve yanındaki edit text’e yazar.

Bir zaman seçici popup’ı açar ve yanındaki edit text’e yazar.

1. **OperationType Class**

İşlem tipleri numaralandırılmıştır.

ekran görüntüsü içeren bir resim

Çok yüksek güvenilirlikle oluşturulmuş açıklama