BEYKENT ÜNİVERSİTESİ
MİMARLIK-MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

MOBİL UYGULAMA PROGRAMLAMA FİNAL PROJE RAPORU

Danışman: Dr. Öğr. Üyesi Atınç YILMAZ

Projeyi Hazırlayan
Mehmet Oğuzhan Akçasoy, 150320099

İstanbul, 2020

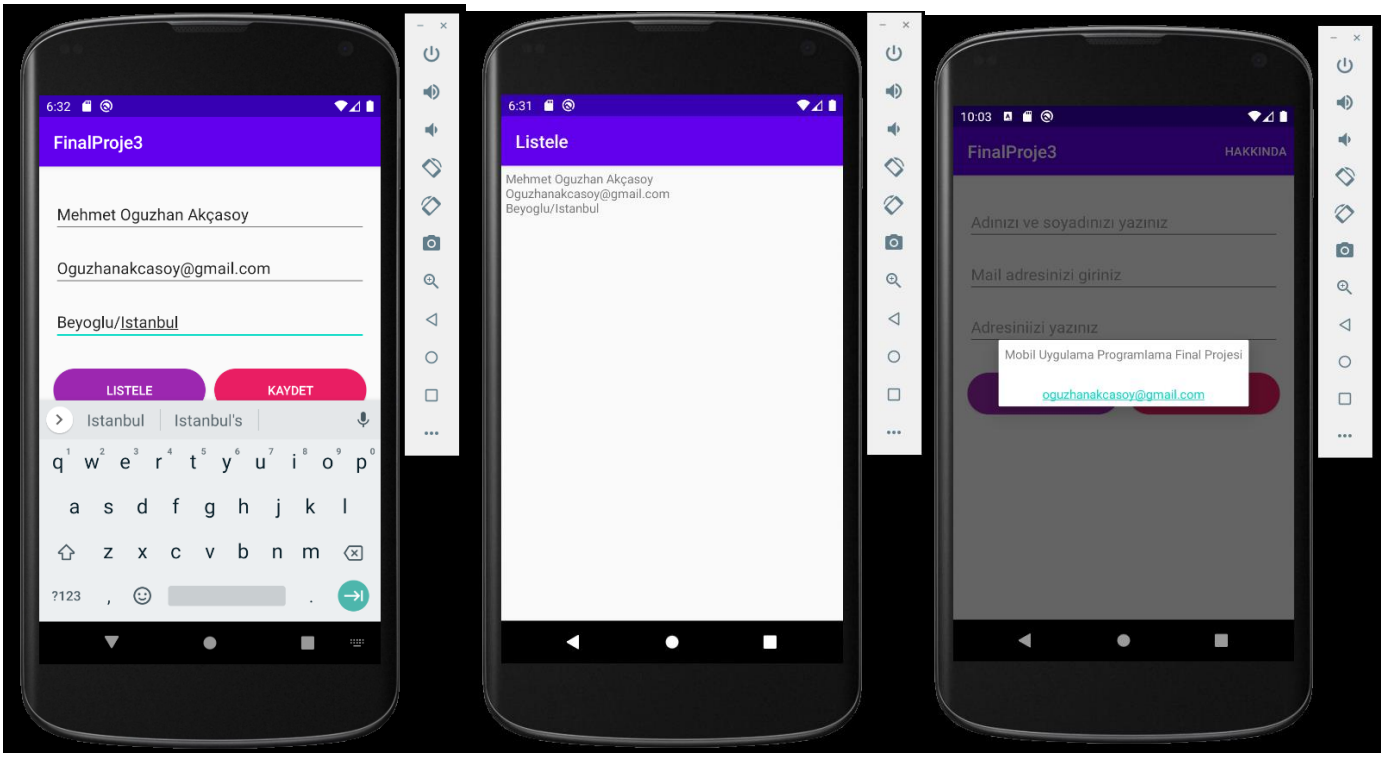
İÇİNDEKİLER

1. GİRİŞ

Mobil uygulama programlama dersi final projesi için yapılan çalışmada; kullanıcıların öğrencilerin ad, soyad, mail adresi ve adres bilgilerini girdikleri ve bu bilgilerin veri tabanına kayıt olmasıyla oluşmaktadır. Bu verilerin tuşa basıldığında farklı bir pencerede veri tabanından alınarak gösterilmesi sağlanmaktadır.

1.1 Uygulama Görüntüsü

Ad soyad, mail ve adres bilgilerinin yazılması gerekilen editTextler ve 2 adet buton resimde yer almaktadır. Butonlar sayesinde veri tabanı üzerinde kaydetme ve görüntüleme işlemlerini sağlanmaktadır. Uygulama içerisinde editTextler, butonlar, intent, cursor, SQLite Veritabanı ve hakkında menüsünün bulunduğu bir diyalog ekranından oluşmaktadır.



Şekil 1:Uygulama İçerisinden Resimler

2.LAYOUT

2.1 Activity_Main.xml

Uygulama içerisindeki düzeni sağlamak için ve tasarım oluşturabilmek için xml dosyasını oluşturmaya başladım. İlk olarak Relative Layout içerisine kullanıcını bilgilerinin alt alta yazabilmesini sağlamak için dikey Linear Layout kullandım. Projede ekran tasarımında kullanılacak bütün kontrollerin view sınıfından türetileceği için Linear Layout içerisine ad soyad, mail ve adres bilgileri için editTextlerin eklemesini yaptım.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:id="@+id/linearLayout">

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="textPersonName"
    android:hint="Adınızı ve soyadınızı yazınız"
    android:ems="10"
    android:id="@+id/et_adsoyad"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="15dp" />

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="textEmailAddress"
    android:ems="10"
    android:hint="Mail adresinizi giriniz"
    android:id="@+id/et_mail"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="15dp" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPostalAddress"
        android:ems="10"
        android:id="@+id/et_adres"
        android:hint="Adresiniizi yazınız"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="15dp" />
```

Şekil 2: EditTextlerin Kod Bloğu

2.2 Hakkında.xml

Uygulama içerisinde bir dialog ekranı oluşturabilmek için bir menü oluşturdum. Bu menü içerisine de id ve title olarak daha sonra oluşturduğum hakkında.xml i tanımlamasını yaptım. Uygulamanın sol üzerinde hakkında butonu ile bu projenin adı ve gmail adresimin görüleceği bir ekranın çıkmasını istediğim için bu xml dosyasının sürekli gözükebilmesi için "always" olarak tanımladım. Açılacak ekrandaki yazan metinleri mail adresine gidebilmesi için autolink içerisine e-mail adresini tanımladım. Textlerin id, yükseklik, genişlik ve boyutunu tanımladım.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Mobil Uygulama Programlama Final Projesi"
        android:padding="7dp"
        android:id="@+id/textView"/>
    <TextView
        android:layout_marginTop="15dp"
        android:padding="5dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="oguzhanakcasoy@gmail.com"
        android:gravity="center"
        android:autoLink="email"
        android:id="@+id/textView2"/>
</LinearLayout>
```

Şekil 3:hakkinda.xml için kodlar

3. JAVA

JAVA dosyaları Java programlama dilinde yazılmış kaynak kodunu içerir. Bu sınıflar denilen yapılandırılmış veri türleri, zamanında nesne örneğini oluşturmak için kullanılan bir nesne yönelimli yaklaşım kullanır. JAVA kaynak kodu dosyaları bir Java derleyicisi kullanarak sınıf dosyalarına derlenmektedir.

3.1 MainActivity.java

Programın ana aktivite kaynak dosyasıdır. AppCompatActivity sınıfından MainActivity adlı bir sınıf türetilir. Sınıf oluşturulduğunda çalışan onCreate() fonksiyonunun kod içeriği yeniden yazılır. Önce üst sınıfın metodu çağrılır, sonra setContentView()metodu ile ana aktivite pencere elemanlarını içeren activity_main.xml dosyası çağrılarak program giriş penceresi ekranda gösterilir. Giriş penceresi içerisine editTextleri ekleyerek gösterilmektedir.

```
public class MainActivity extends AppCompatActivity {

    private static final int DIALOG_HAKKINDA=1;
    EditText et_adsoyad, et_mail, et_adres;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //3 satır.
        setContentView(R.layout.activity_main);
        et_adsoyad = (EditText) findViewById(R.id.et_adsoyad);
        et_mail = (EditText) findViewById(R.id.et_mail);
        et_adres = (EditText) findViewById(R.id.et_adres);
    }
```

Şekil 4:MainActivity.java kodları

onCreateOptionsMenu ile menu_main dosyası eklenerek giriş ekranında menü barı gözükmesi sağlanamaktadır. onOptionsItemSelected ile seçildiğinde çalışmasını istediğim dosyanın tanımlaması yapılmıştır. Bu sayede kullanıcı menü içerisinde item seçtiğinde idsi hakkında olan dosya çalışmaktadır.

```
@Override
public boolean onCreateOptionsMenu(Menu menu){
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item){
    int id = item.getItemId();
   switch (id){
        case R.id.hakkinda:
            showDialog(DIALOG_HAKKINDA);
        return true;
        default:
            return super.onOptionsItemSelected(item);
   }
}
```

Şekil 5:Hakkında menüsünün kodu

Ekranının alt tarafında bulunan listele ve kaydet butonun tıklandığında öğrenci nesnelerini ogrenci.java dosyasından ve veri tabanına ekleme yapabilmesi sağlamak için butonaDokunuldu isminde bir yöntem oluşturuldu. Ad soyad, mail ve adres bilgilerini öğrenci classında getirilmesi ve string olarak kullanılması sağlanmaktadır. Alınan bu veriler veri tabanı içerisindeki kayıt ekle metoduna gönderilmektedir. Bu girilen bilgilerden hatalı (-1) geri değer alınırsa hata mesajı, başarılı ise işlem başarılı mesajı yayınlanması sağlanmaktadır.

Listele butonu tıklandığında Listele.classı ile bir intent, niyet veya yeni bir ekran oluşturulmaktadır. Daha sonra ise bu ekran startActivity ile listele.class ile başlatılmaktadır.

```
    @Override
    protected Dialog onCreateDialog(int id){
        Dialog dialog = null;
        switch (id){
            case DIALOG_HAKKINDA:
                dialog=new Dialog(MainActivity.this);
                dialog.setTitle("Hakkında");
                dialog.setContentView(R.layout.hakkinda);
                break;
            default:
                dialog=null;
        }
        return dialog;
    }
    public void butonaDokunuldu(View view) {
        switch (view.getId()) {
            case R.id.btn_kaydet:
                String adsoyad = et_adsoyad.getText().toString();
                String mail = et_mail.getText().toString();
                String adres = et_adres.getText().toString();
                Ogrenci ogrenci = new Ogrenci(adsoyad, mail, adres);
                Veritabani db= new Veritabani(getApplicationContext());
                long id=db.KayıtEkle(ogrenci);
                if(id==-1){
                    Toast.makeText(MainActivity.this, "Hay aksi! Kayıt işleminde bir
hata oluştu..", Toast.LENGTH_SHORT).show();
                }else{
                    Toast.makeText(getApplicationContext(),"Kayıt işlemi
başarılı",Toast.LENGTH_LONG).show();
                }
                et_adres.setText("");
                et_adsoyad.setText("");
                et_mail.setText("");
                break;
            case R.id.btn_listele:
                Intent intent = new Intent(getApplicationContext(),Listele.class);
                startActivity(intent);
        }
    }
}
```

Şekil 6: Butona Dokunulduğunda gerçekleşen işlemler için kodlar

3.2 Ogrenci.java

MainActivity içerisine alınan verileri veri tabanına kaydetmem gerikiyor. Bu nedenle ogrenci.java ve veritabanı.java isimli 2 yeni sınıf oluşturdum. ogrenci sınıfının kendisine ait özellikleri olması gerektiği için private olarak ve string türünde ad soyad, mail ve adresi yapıcı metot ile tanımladım. Bu sayede yapıcı metod ile parametreler atanmış oldu.

Öğrenci nesnesinin mainactivity içerisine tanımlasını yaptım. Generate ile getter ve setter metotlarını oluşturdum. Bu yöntem ile set ile değer ataması ve get ile değerleri ogrenci sınıfına getirmiş oldum.

```
public class Ogrenci {

    private String AdSoyad;
    private String Mail;
    private String Adres;

    public Ogrenci() {
    }
    public Ogrenci(String adSoyad, String mail, String adres) {
        AdSoyad = adSoyad;
        Mail = mail;
        Adres = adres;
    }
    public void setAdSoyad(String adSoyad) {

        AdSoyad = adSoyad;
    }
    public void setMail(String mail) {

        Mail = mail;
    }
    public void setAdres(String adres) {

        Adres = adres;
    }
    public String getAdSoyad() {

        return AdSoyad;
    }
    public String getMail() {

        return Mail;
    }
    public String getAdres() {

        return Adres;
    }
}
```

Şekil 7:Ogrenci sınıfı ile Getter ve Setter metotları

3.3 Veritabanı.java

Veritabanı oluşturmak için veritabanı.java isimli bir sınıf oluşturdum. Private ile sabitleri tanımlıyorum. String türünde veritabanı adı, tablo adı, veri tabanı versiyonu, ad, mail, adres ve id sabitlerini tanımlamasını yaptım.

```
public class Veritabani extends SQLiteOpenHelper {
    private static final String DATABASE_NAME = "ogrenci_veritabani";
    private static final String TABLE_NAME = "ogrenci_tablosu";
    private static final int DATABASE_VERSION = 1;

    private static final String AD = "ad_soyad";
    private static final String MAIL = "mail";
    private static final String ADRES = "adres";
    private static final String ID = "_id";

    public Veritabani(Context context) {

        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
```

Şekil 8: Veritabanı.java kodları

Bir veritabanı oluşturabilmemiz için farklı bir sınıftan kalıtım (extends) almamız gerekiyor. Bu sınıfın ismi SQLiteOpenHelperdir. Bu sınıfın yapıcı ve 2 farklı metodu vardır. SQLiteOpenHelperin çalışabilmesi için onCreate ve onUpgrade metodlarını implement etmek gerekiyor. Bu metodların tanımından sonra yapıcı metodu ekliyorum ve içerisine veri tabanı adı ve veri tabanı versiyonunu tanımlıyorum.

```
@Override
public void onCreate(SQLiteDatabase db) {

    String tablo_olustur = "CREATE TABLE " + TABLE_NAME +
            " (" + ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
            AD + " TEXT, " +
            MAIL + " TEXT, " +
            ADRES + " TEXT);";

    db.execSQL(tablo_olustur);
}
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
    onCreate(db);
}
```

Şekil 9: SQLiteOpenHelper ve Metotları

Veri tabanı oluştururken SQLiteBrowser uygulamasını kullanabilir veya kendimiz de yazabiliriz. İlk olarak onCreate içiresine tablo oluştur ile daha önce tanımladığımız sabitleri Text veya integer olmasına göre tanımlıyoruz. ID benzersiz bir değer olacağı için "Integer Primary Key Autoincrement" olacaktır. db.execSQL komutu sayesinde oluşturduğum Sqlite sorgusunu veri tabanında çalıştıracak. onUpgrade metot içerisine "Drop Table İf Exists+Table_Name" ile veritabanı güncellendikçe silinir ve onCreate(db) metodu tekrar döndürülür.

```
@Override
public void onCreate(SQLiteDatabase db) {

    String tablo_olustur = "CREATE TABLE " + TABLE_NAME +
            " (" + ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
            AD + " TEXT, " +
            MAIL + " TEXT, " +
            ADRES + " TEXT);";

    db.execSQL(tablo_olustur);
}
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
    onCreate(db);
}
```

Şekil 10: onCreate ve onUpgrade

Veri tabanı ve Ogrenci sınıfını tanımlaması yapıldıktan sonra bu metodlara işlem yaptırabilmek için KayıtEkle ile parametre olarak ogrenci nesnesinin tanımlaması yapılmıştır. Kayıt ekle metodu veri tabanını metot olarak tanımlamasını KayıtEkle metoduna tanımladım. Bu tanımlama içerisine verinin yazılabilmesini sağlamak için WritableDatabase metodunu ekledim.

Kayıt ekle ve güncelleme işlemleri için contentValues sınıfından yararlanmak gerekir. Tanımlanan metodun içerisine put ile cv olarak tanımadığım content values metoduna parametrelerin ataması yapılır. Burada kolon isimleri parametre olarak verilmektedir. Öğrenci bilgilerini alacağım sınıf olan "Ogrenci" sınıfından parametreleri getAdsoyad, getMail, getAdres ile verileri getirebiliriz. Buradaki bilgiler ise satırdır.

```
public long KayıtEkle(Ogrenci ogrenci) {
    SQLiteDatabase db = this.getWritableDatabase();

    ContentValues cv = new ContentValues();
    cv.put(AD, ogrenci.getAdSoyad());
    cv.put(MAIL, ogrenci.getMail());
    cv.put(ADRES, ogrenci.getAdres());
```

Şekil 11: KayıtEkle Metot Kodları

Bu yöntemler ile elde ettiğim verileri veri tabanına eklemek gerekiyor. İnsert metodu ile long türünde bir değer döndürür. Bu değer eklenen satırın idsidir. Ekleme sırasında bir hata meydana gelmişse dönen hata -1 olur. Aslında public void olan metodu geriye değer döndürmez fakat burada döndürülmesi ile hata tespiti yapılabilir. Metodu public longa çeviriyorum. Long id inserte tanımlamasını yapılır. İd değeri dönebilmesi için return komutunu yazıyorum.

```
    Long id = db.insert(TABLE_NAME, null, cv);
    db.close();
    return id;
}
```

Şekil 12: Değer Tanımalama

Döndürülen id değerini MainActivitydeki btn_kaydet sınıfına tanımlıyorum. Geriye dönen değeri alabilmemi sağlamaktadır.

```
       if(id==-1){
    Toast.makeText(MainActivity.this, "Hay aksi! Kayıt işleminde bir hata oluştu..",
Toast.LENGTH_SHORT).show();
}else{
    Toast.makeText(getApplicationContext(),"Kayıt   işlemi
başarılı",Toast.LENGTH_LONG).show();
}
```

Şekil 13: Başarılı ve Hata Mesajı

-1 değeri dönerse veya dönmez ise bir mesaj yayınlatabiliriz. Hata mesajı ise kısa başarılı ise uzun göstermesi için Lengthyi ayarlıyorum. Bu aşamada Kullanıcı bilgileri doldurduktan sonra editTextlerde bilgilerin kaldığını fark ettim ve temizlenmesi için setText(«») oluşturdum.

```
et_adres.setText("");
et_adsoyad.setText("");
et_mail.setText("");
```

Şekil 14: EditTextlerin Silinmesi

3.4 Listele.java

Listele için Class oluşturup xml dosyalarını ayrı oluşturmak yerine, blank activity sayesinde java ve xmlini aynı anda oluşturup manifestte tanımlamasını yaptım. Veri tabanından aldığımız verileri listele içerisinde göstereceğim için textviewi listele.java içerisine tanımladım.

```
tv= (TextView) findViewById(R.id.tv);
```

Uygulamada listele butonuna tıklandığında listele aktivitesine yönlendirmem gerekiyor. Mainactivitye içerisine case R.id.btn_listele nin içerisine intent tanımlıyorum ve android.context.intenti projeye import ettim. İçerisine getApplicationContext ve listele.class parametrelerini tanımladım.

```
Intent intent = new Intent(getApplicationContext(),Listele.class);
startActivity(intent);
```

Listele.java içerisine veri tabanından bir nesne oluşturuyorum. List olarak nesne tanımladım. (nesne yönelimli programlama). Dizilerle dinamik olarak arttırım, nesne veya sınıf yapılamadığı için liste kullandım. Veri tabanından getireceğim verileri öğrenciliste depolayacağım. Bu nedenle veri tabanı içerisine tüm kayıtları getir metodunu oluşturdum.

```
Veritabani db=new Veritabani(getApplicationContext());
List<Ogrenci> ogrenciList=new ArrayList<Ogrenci>();
```

```
Listele.java içerisinde; ogrenciList=db.TumKayitlariGetir();

Veritabanı.java içerisinde; public List<Ogrenci> TumKayitlariGetir() {}
```

3.5 Cursor

Veritabanı.java içerisine SQLitedatabase db= getReadabledatabase yazdım. Verileri okuyabilmek için sütunlar isimde bir string oluşturdum ve içerisine kolonların isimlerini yazdım. (AD, MAİL, ADRES, ID). Daha sonra cursor oluşturdum. Cursor içerisine tablo ismi ve sütunları yazıyorum. Cursorun satır satır dolaşacağı zaman ad, mail, adres sütunundaki verileri alabilmesi için integer türünde bir index numarasına ihtiyacı vardır. İnt olarak tanımlamaları ve kolon isimleri ile çağırdım.

```
public List<Ogrenci> TumKayitlariGetir() {
    SQLiteDatabase db = this.getReadableDatabase();
    String[] sutunlar = new String[]{AD, MAIL, ADRES, ID};
    Cursor c = db.query(TABLE_NAME, sutunlar, null, null, null, null, null);
    ...
return ogrenciList
```

Cursorun satır satır dolaştığı zaman indexe ihtiyacı vardır.

```
int adsirano = c.getColumnIndex(AD);
int mailsirano = c.getColumnIndex(MAIL);
int adresssirano = c.getColumnIndex(ADRES);
```

İndexler integer türünde tanımlaması yapılır. Ogrenci sınıfında bir arrayllist oluşturuyorum ve for döngüsü kullanıyorum. For içerisine cursor metodlarını tanımlarız. Burada ilk kayıta giden cursor moveToFirst metodunu daha sonrasında bütün kayıtları dolaşması için AfterLast metodunu tanımlıyorum. Cursor bir sonraki kayıta götürmek için moveToNext komutunu for içerisine tanımlıyoruz. Ogrenci nesnesi oluşturuyoruz ve değerlerini atıyoruz. Ogrenci metoduna int değerlerini set ediyoruz. Bu sayede her döngüde öğrenci nesnesine değerleri atamış oluyoruz. Bu değerleri ogrenciliste ekliyoruz. Veritabanını kapatıyoruz ve return ile tekrarını sağlıyoruz.

Listele.java içerisine veri eklemek ve performans iyileştirmesi için str yerine stringbuilder kullanıyorum. Ogrenci verilerini alt alta gelecek şekilde tanımlıyorum. Yeni eklenecek veriler için 2 satır alta yazmasını sağlıyorum. Son olarak textViewe set ediyorum.

```
public class Listele extends AppCompatActivity {
   // private ActionMode actionMode;
    private Long id;

    TextView tv;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_listele);

        try{
            tv= (TextView) findViewById(R.id.tv);
            Veritabani db=new Veritabani(getApplicationContext());
            List<Ogrenci> ogrenciList=new ArrayList<Ogrenci>();

            ogrenciList=db.TumKayitlariGetir();

            StringBuilder stringBuilder=new StringBuilder();
            for(final Ogrenci ogrenci:ogrenciList){

stringBuilder.append(ogrenci.getAdSoyad()+"\n"+ogrenci.getMail()+"\n"+ogrenci.getAdre
s()+"\n\n");
    }
            tv.setText(stringBuilder);
        }
        catch (Exception e){
            tv.setText("Herhangi bir kayıt bulunamadı..");

        }
}}
```

4. DRAWABLE

Drawable içerisine 2 buton için tasarım oluşturuyorum. kaydet.xml ve listele.xml. android:shape=«rectangle» olacak şekilde atamasını yaptım.

https://material.io/design/color/the-color-system.html#tools-for-picking-colors sitesinden yandaki resimdeki gibi 2 farklı renk seçiyorum ve android:color=«»şeklinde butonların renklerini tanımladım. Butonların köşelerinin daha güzel gözükmesi için corners içerisine android:radius=«25dp» olarak tanımladım. Bu sayede köşelerden kırpma işlemi yaptırmış oldum.

Red 50	#FFEBEE
100	#FFCDD2
200	#EF9A9A
300	#E57373
400	#EF5350
500	#F44336
600	#E53935
700	#D32F2F
800	#C62828
900	#B71C1C
A100	#FF8A80
A200	#FF5252
A400	#FF1744
A700	#D50000

Pink 50	#FCE4EC
100	#F8BBD0
200	#F48FB1
300	#F06292
400	#EC407A
500	#E91E63
600	#D81B60
700	#C2185B
800	#AD1457
900	#880E4F
A100	#FF80AB
A200	#FF4081
A400	#F50057
A700	#C51162

Purple 50	#F3E5F5
100	#E1BEE7
200	#CE93D8
300	#BA68C8
400	#AB47BC
500	#9C27B0
600	#8E24AA
700	#7B1FA2
800	#6A1B9A
900	#4A148C
A100	#EA80FC
A200	#E040FB
A400	#D500F9
A700	#AA00FF

4.1 Kaydet.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
    <solid
        android:color="#E91E63"/>
    <corners
        android:radius="25dp"/>
</shape>
```

4.1.1 Kaydet_press.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">

    <solid
        android:color="#880E4F"/>
    <corners
        android:radius="25dp"/>

</shape>
```

4.1.2 Kaydet_selector.xml

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:drawable="@drawable/kaydet_press"
        android:state_pressed="true"/>
    <item android:drawable="@drawable/kaydet"/>
</selector>
```

4.2 Listele.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
    <solid
        android:color="#9C27B0"/>
    <corners
        android:radius="25dp"/>
</shape>
```

4.2.1 Listele_ press.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
```

```
android:shape="rectangle">
    <solid
        android:color="#4A148C"/>
    <corners
        android:radius="25dp"/>
</shape>
```

4.2.2 Listele_ selector.xml

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:drawable="@drawable/listele_press"
        android:state_pressed="true"/>
    <item android:drawable="@drawable/listele"/>

</selecto
```

5. MENU

5.1 Contex_menu.xml

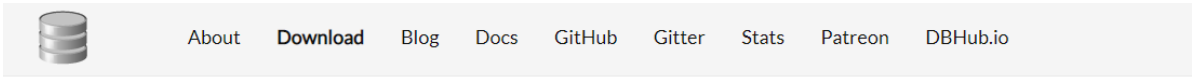
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".MainActivity">
    <item
        android:id="@+id/sil"
        android:orderInCategory="0"
        android:icon="@android:drawable/ic_menu_delete"
        android:title="sil"
        app:showAsAction="always">
    </item>

    <item
        android:id="@+id/düzenle"
        android:orderInCategory="1"
        android:icon="@android:drawable/ic_menu_edit"
        android:title="Güncelle"
        app:showAsAction="always">
    </item>
</menu>
```

5.2 Menu_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
```

```
    tools:context=".MainActivity">
    <item
        android:id="@+id/hakkinda"
        android:orderInCategory="0"
        android:title="Hakkında"
        app:showAsAction="always">
    </item>
</menu>
```

6. SQLite Browser ile Veri Tabanı Görüntüleme



Downloads

(**Please** consider sponsoring us on Patreon 😄)

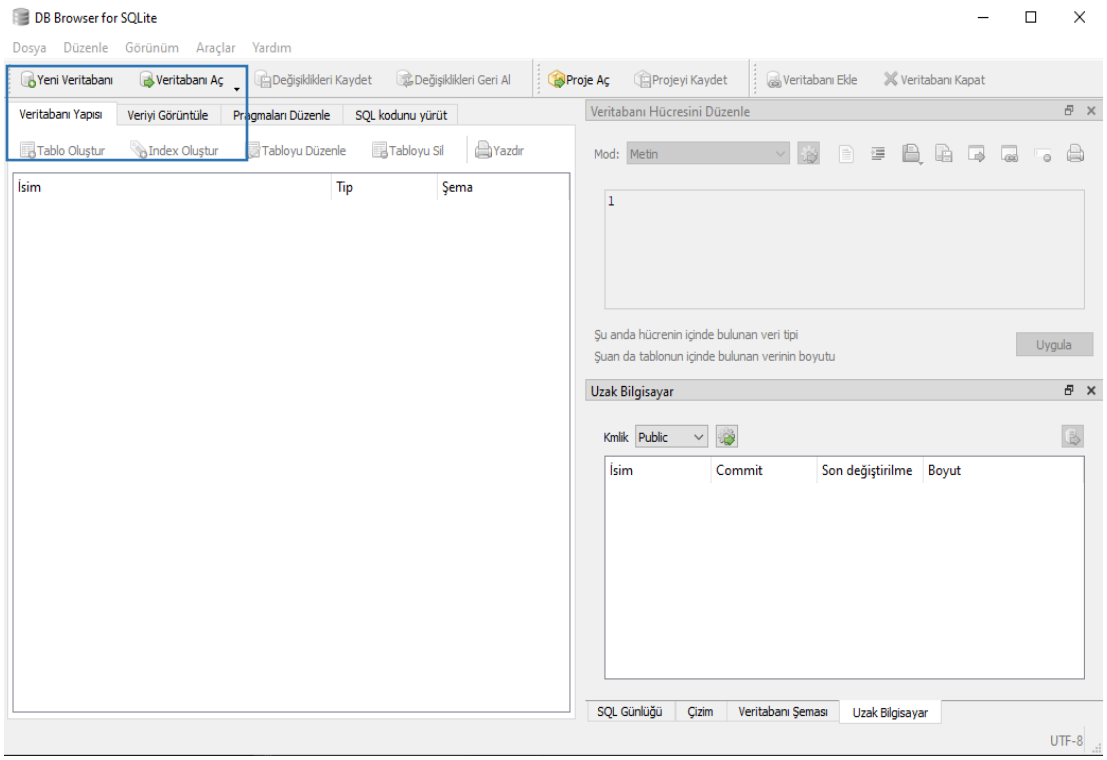
Windows

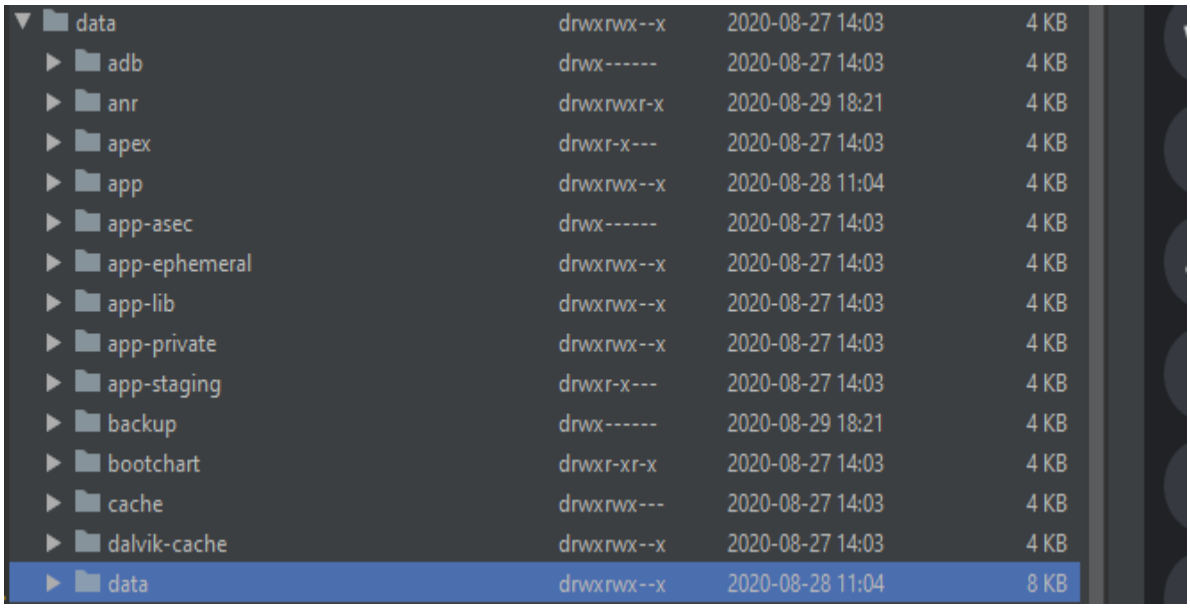
Our latest release (3.12.0) for Windows:

- DB Browser for SQLite - Standard installer for 32-bit Windows
- DB Browser for SQLite - .zip (no installer) for 32-bit Windows
- DB Browser for SQLite - Standard installer for 64-bit Windows
- DB Browser for SQLite - .zip (no installer) for 64-bit Windows
- DB Browser for SQLite - PortableApp

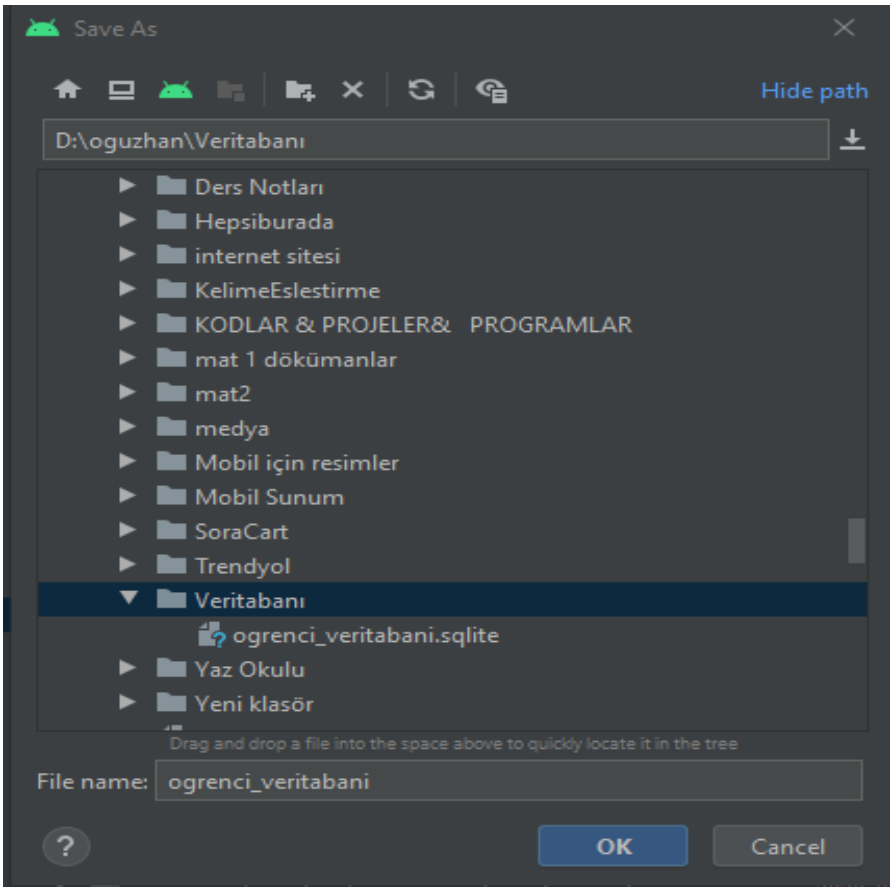
Note - If for any reason the standard Windows release does not work (e.g. gives an error), try a nightly build (below).

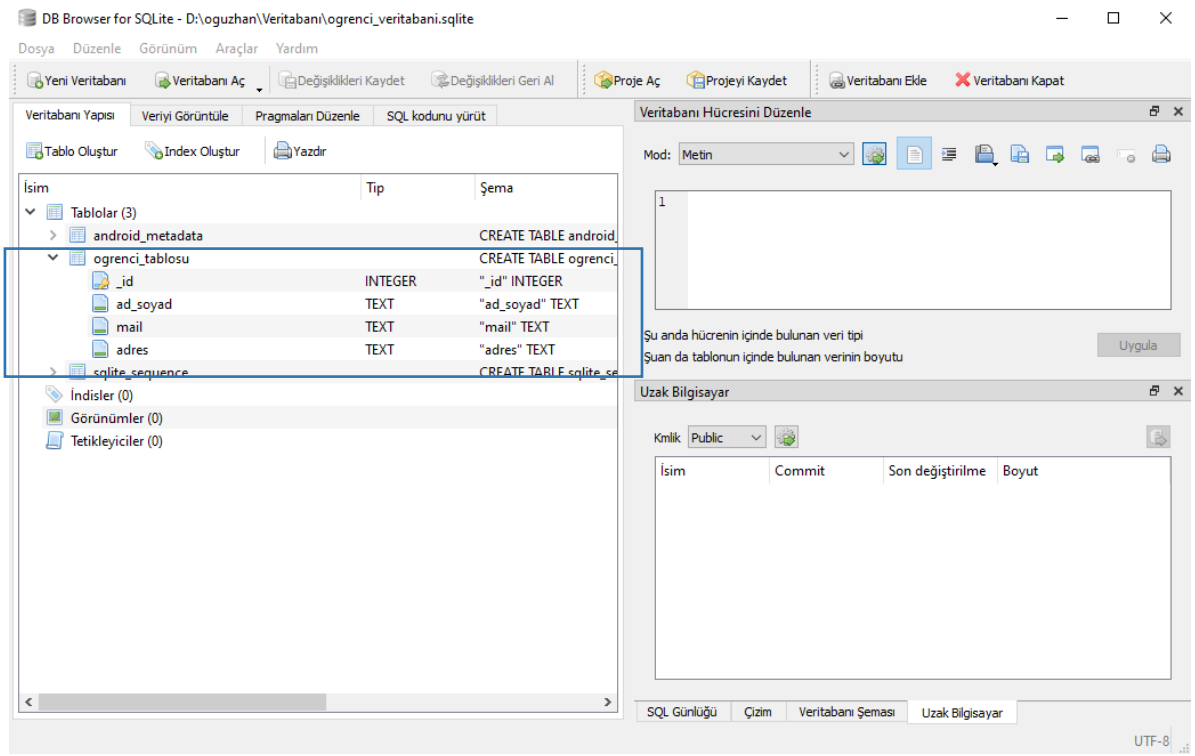
Nightly builds often fix bugs reported after the last release. 😄

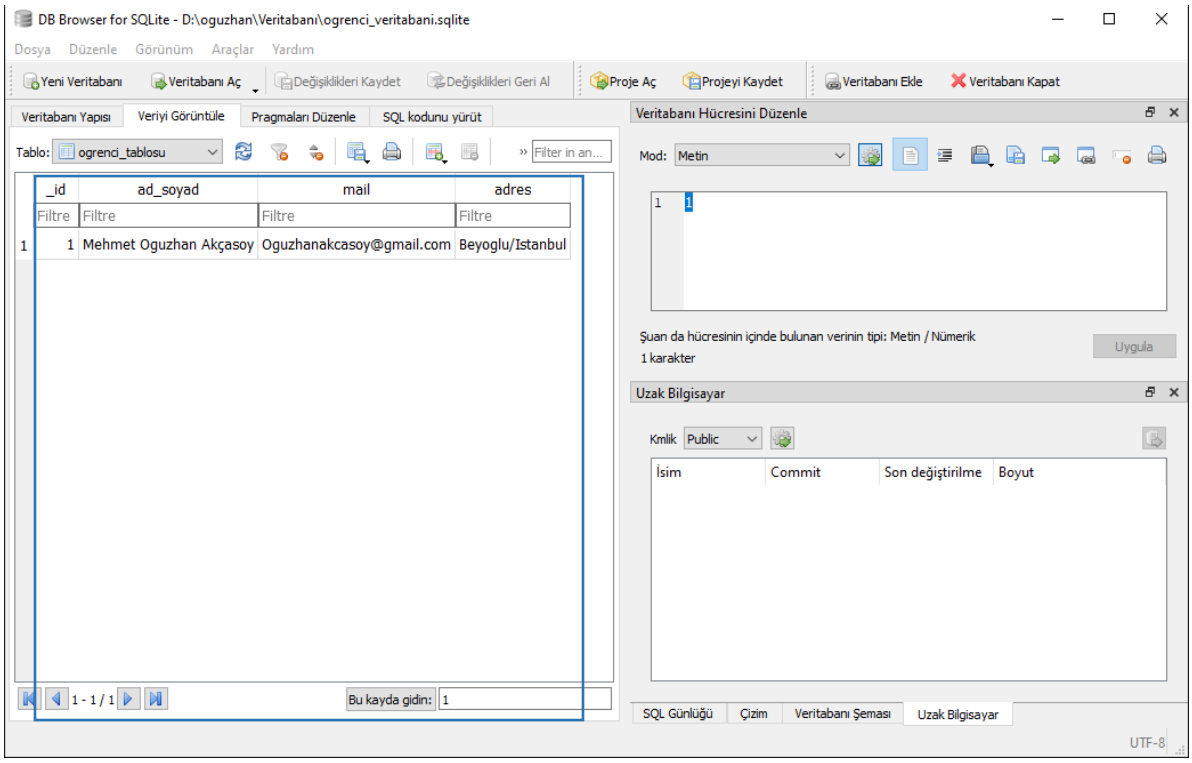

DB Browser for SQLite
Dosya Düzenle Görünüm Araçlar Yardım
Yeni Veritabanı
Veritabanı Aç
Değişiklikleri Kaydet
Değişiklikleri Geri Al
Proje Aç
Projeyi Kaydet
Veritabanı Ekle
Veritabanı Kapat
Veritabanı Yapısı
Veriyi Görüntüle
Pragmaları Düzenle
SQL kodunu yürüt
Tablo Oluştur
Index Oluştur
Tabloyu Düzenle
Tabloyu Sil
Yazdır
İsim
Tip
Şema
Veritabanı Hücresini Düzenle
Mod: Metin
1
Şu anda hücrenin içinde bulunan veri tipi
Şuan da tablonun içinde bulunan verinin boyutu
Uygula
Uzak Bilgisayar
Kmlik Public
İsim
Commit
Son değiştirilme
Boyut
SQL Günlüğü
Çizim
Veritabanı Şeması
Uzak Bilgisayar
UTF-8


data drwxrwx--x 2020-08-27 14:03 4 KB
adb drwx------ 2020-08-27 14:03 4 KB
anr drwxrwxr-x 2020-08-29 18:21 4 KB
apex drwxr-x--- 2020-08-27 14:03 4 KB
app drwxrwx--x 2020-08-28 11:04 4 KB
app-asec drwx------ 2020-08-27 14:03 4 KB
app-ephemeral drwxrwx--x 2020-08-27 14:03 4 KB
app-lib drwxrwx--x 2020-08-27 14:03 4 KB
app-private drwxrwx--x 2020-08-27 14:03 4 KB
app-staging drwxr-x--- 2020-08-27 14:03 4 KB
backup drwx------ 2020-08-29 18:21 4 KB
bootchart drwxr-xr-x 2020-08-27 14:03 4 KB
cache drwxrwx--- 2020-08-27 14:03 4 KB
dalvik-cache drwxrwx--x 2020-08-27 14:03 4 KB
data drwxrwx--x 2020-08-28 11:04 8 KB


com.example.finalproje3 drwx------ 2020-08-28 11:05 4 KB
cache drwxrws--x 2020-08-28 11:04 4 KB
code_cache drwxrws--x 2020-08-28 11:04 4 KB
databases drwxrwx--x 2020-08-28 11:05 4 KB
ogrenci_veritabani -rw-rw---- 2020-08-28 11:05 20 KB
ogrenci_veritabani-journal -rw-rw---- 2020-08-28 11:05 0 B


Save As
Hide path
D:\oguzhan\Veritabanı
Ders Notları
Hepsiburada
internet sitesi
KelimeEslestirme
KODLAR & PROJELER& PROGRAMLAR
mat 1 dökümanlar
mat2
medya
Mobil için resimler
Mobil Sunum
SoraCart
Trendyol
Veritabanı
ogrenci_veritabani.sqlite
Yaz Okulu
Yeni klasör
Drag and drop a file into the space above to quickly locate it in the tree
File name:
ogrenci_veritabani
OK
Cancel


DB Browser for SQLite - D:\oguzhan\Veritabanı\ogrenci_veritabani.sqlite
Dosya Düzenle Görünüm Araçlar Yardım
Yeni Veritabanı
Veritabanı Aç
Değişiklikleri Kaydet
Değişiklikleri Geri Al
Proje Aç
Projeyi Kaydet
Veritabanı Ekle
Veritabanı Kapat
Veritabanı Yapısı
Veriyi Görüntüle
Pragmaları Düzenle
SQL kodunu yürüt
Tablo Oluştur
Index Oluştur
Yazdır
İsim
Tip
Şema
Tablolar (3)
android_metadata
CREATE TABLE android_
ogrenci_tablosu
CREATE TABLE ogrenci_
_id
INTEGER
"_id" INTEGER
ad_soyad
TEXT
"ad_soyad" TEXT
mail
TEXT
"mail" TEXT
adres
TEXT
"adres" TEXT
sqlite_sequence
CREATE TABLE sqlite_se
İndisler (0)
Görünümler (0)
Tetikleyiciler (0)
Veritabanı Hücresini Düzenle
Mod:
Metin
1
Şu anda hücrenin içinde bulunan veri tipi
Şuan da tablonun içinde bulunan verinin boyutu
Uygula
Uzak Bilgisayar
Kmlik
Public
İsim
Commit
Son değiştirilme
Boyut
SQL Günlüğü
Çizim
Veritabanı Şeması
Uzak Bilgisayar
UTF-8


DB Browser for SQLite - D:\oguzhan\Veritabanı\ogrenci_veritabani.sqlite
Dosya
Düzenle
Görünüm
Araçlar
Yardım
Yeni Veritabanı
Veritabanı Aç
Değişiklikleri Kaydet
Değişiklikleri Geri Al
Proje Aç
Projeyi Kaydet
Veritabanı Ekle
Veritabanı Kapat
Veritabanı Yapısı
Veriyi Görüntüle
Pragmaları Düzenle
SQL kodunu yürüt
Tablo:
ogrenci_tablosu
Filter in an...
_id
ad_soyad
mail
adres
Filtre
1
Mehmet Oguzhan Akçasoy
Oguzhanakcasoy@gmail.com
Beyoglu/Istanbul
1 - 1 / 1
Bu kayda gidin:
Veritabanı Hücresini Düzenle
Mod:
Metin
Şuan da hücresinin içinde bulunan verinin tipi: Metin / Nümerik
1 karakter
Uygula
Uzak Bilgisayar
Kmlik
Public
İsim
Commit
Son değiştirilme
Boyut
SQL Günlüğü
Çizim
Veritabanı Şeması
Uzak Bilgisayar
UTF-8