Móviles.
Tema: Fecha y Hora.

Introducción.
En Java, las fechas y horas están almacenadas en varias formas, dependiendo de las necesidades. El tipo `long` es un tipo de datos primitivos capaz de almacenar el número de milisegundos transcurridos desde un punto específico en el tiempo (tiempo Unix).

La clase `Date(java.util.Date)` es igual clase pública para almacenar la fecha y la hora en una forma que puede ser razonablemente manipulada sin tener que pensar constantemente sobre el tiempo en términos de milisegundos. La clase `Calendar(java.util.Calendar)` es una clase pública para trabajar con diferentes calendarios, así como también para manipular la información de la fecha y la hora en una variedad de formas. La clase `GregorianCalendar` (una subclase de `java.util.Calendar`) es usada, principalmente, para la manipulación de la fecha en el hemisferio Oeste, en donde usamos un calendario de 12 meses, con 7 días a la semana y dos épocas (BC y AD). Se puede determinar la fecha y la hora usando un método estático proporcionado por la clase `System (java.lang.System)`:

```
long msTime = System.currentTimeMillis();
Date curDateTime = new Date(msTime);
```

Otra forma para determinar la fecha y la hora actual es usando el constructor por defecto para la clase `Date`, el cual crea un objeto `Date` con la fecha y la hora actual.

```
Date anotherCurDate = new Date();
```

Por ejemplo, con la clase `Calendar`:

```
GregorianCalendar bday = new GregorianCalendar(1977, Calendar.APRIL, 12);
```

Se pueden usar objetos calendarios para manipular las fechas y extraer la información interesante sobre ellas. Por ejemplo, se puede usar el método `get()` para determinar el día de la semana en el que se nació:

```
int dayOfWeek=bday.get(Calendar.DAY_OF_WEEK);  // Regresa 3, for Tuesday
```

Observar que los meses del calendario se basan en el 0, por lo que Enero es el mes 0 no 1, Febrero es el mes 1, Marzo el 2, Abril es el mes 3 no el 4. Se puede extraer el objeto `Date` desde una configuración de fecha `Calendar` usando el método `getTime()`.

Especificar el formato con el tipo `String`, usando los códigos apropiados para los diferentes bits de información (ver la documentación en detalle de `SimpleDateFormat` o códigos individuales). Luego, aplicar esa información para una fecha y hora específica. Por ejemplo:

```
Date anotherCurDate = new Date();
SimpleDateFormat formatter = new SimpleDateFormat("EEEE, MMMM d 'at' hh:mm a 'in the year'
yyyy G");
String formattedDateString = formatter.format(anotherCurDate);
```

DESARROLLO.

EJEMPLO 1.
Paso 1. Crear un nuevo proyecto y modificar el archivo MainActivity.java con el código siguiente.

```
// MainActivity.java
import android.app.*;
import android.os.*;
import android.view.*;
import android.view.View.OnClickListener;
import android.widget.*;
import java.util.*;
public class MainActivity extends Activity implements OnClickListener {
```

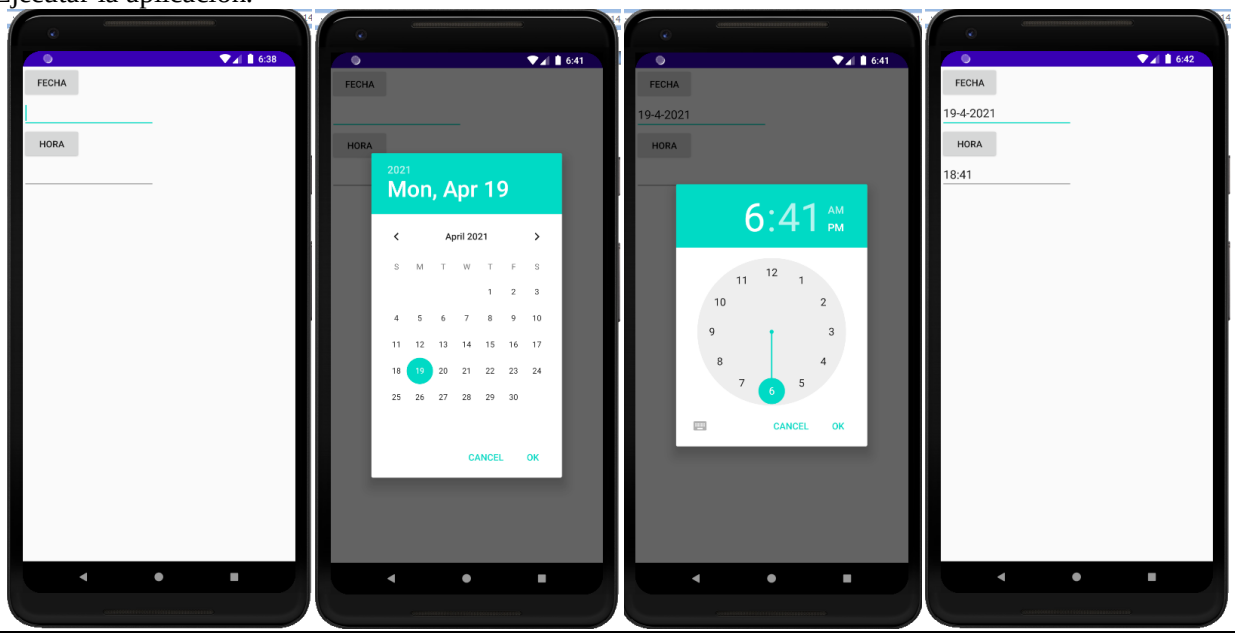
```
    Button      jbnF, jbnH;
    EditText    txtDate, txtTime;
    int         a, m, d, h, n;
    @Override
    protected void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.activity_main);
        jbnF = (Button)findViewById(R.id.xbnF); jbnF.setOnClickListener(this);
        jbnH = (Button)findViewById(R.id.xbnH); jbnH.setOnClickListener(this);
        txtDate = (EditText)findViewById(R.id.xetF);
        txtTime = (EditText)findViewById(R.id.xetH);
    }
    @Override
    public void onClick(View v) {
        if (v == jbnF) {
            Calendar c = Calendar.getInstance();
            a = c.get(Calendar.YEAR);
            m = c.get(Calendar.MONTH);
            d = c.get(Calendar.DAY_OF_MONTH);
            DatePickerDialog dpd = new DatePickerDialog(this, new
DatePickerDialog.OnDateSetListener() {
                        @Override
                        public void onDateSet(DatePicker dp, int ye, int mo, int di) {
                            txtDate.setText(di + "-" + (mo + 1) + "-" + ye);
                        }
                    }, a, m, d);
            dpd.show();
        }
        if (v == jbnH) {
            Calendar c = Calendar.getInstance();
            h = c.get(Calendar.HOUR_OF_DAY);
            n = c.get(Calendar.MINUTE);
            TimePickerDialog tpd = new TimePickerDialog(this, new
TimePickerDialog.OnTimeSetListener() {
                        @Override
                        public void onTimeSet(TimePicker tp, int ho, int mi) {
                            txtTime.setText(ho + ":" + mi);
                        }
                    }, h, n, false);
            tpd.show();
        }
    }
}
```

Paso 2. Abrir el archive activity_main.xml y modificarlo con el código siguiente.

```
// activity_main.xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <Button
        android:id="@+id/xbnF"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:text="Fecha" />
    <EditText
        android:id="@+id/xetF"
        android:layout_width="200dp"
        android:layout_height="wrap_content" />
    <Button
        android:id="@+id/xbnH"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hora" />
    <EditText
        android:id="@+id/xetH"
        android:layout_width="200dp"
        android:layout_height="wrap_content" />
</LinearLayout>
```

Paso 3. Ejecutar la aplicación.



Ejercicio 1.
Diseñar una aplicación que permita guardar las fechas y horarios de nacimiento de al menos cinco usuarios en una base de datos.

Nota. Guardar las imágenes de las aplicaciones en un documento AlumnoFechaHoraGrupo.pdf y enviarlo al sitio indicado por el profesor.