

## Android：データを保存する方法

Android のアプリケーションで、データを保存する方法を説明します。

保存する方法としては以下のものがあります。

- ファイルとして保存
  - Preference
  - データベース (SQLite)
- 

### ●ファイルへ書き込む

Android のファイルへの書き出しはアクセス権限の設定部分があるので読み込みの `openFileInput` メソッドより、引数が増えています。

- `public abstract FileOutputStream openFileOutput`

`(String name, int mode)`

をつかいます。まずは、出力用ストリーム (`OutputStream`) インスタンスを、`openFileOutput()` メソッドで取得します。

このとき第一引数で指定するファイル名は、**パス無しでファイル名のみを指定**します。ファイルパスは、OS により、Android アプリ毎に一意に決められますので、アプリ側からパスを指定する事は出来ません。

第二引数は操作モードです。

以下の定数を指定します。

`MODE_APPEND`：既存ファイルを追記モードで開きます。

`MODE_PRIVATE`：ローカルファイルを作成した Android アプリのみにアクセス許可を与える場合に指定します。

`MODE_WORLD_READABLE`：他アプリから読み取り可能にします。

`MODE_WORLD_WRITEABLE`：他アプリから書込み可能にします。

複数指定したい場合は、上記定数を論理和で指定します。

そして、取得した `OutputStream` インスタンスを使い、文字コードは UTF-8 として、`PrintWriter` インスタンスを生成して、あとは `PrintWriter.append()` で書き込みたい文字列を渡して、`close()` で保存する、というだけです。

---

## ● ファイルの読み込み方法

それでは、保存されているローカルファイルを読み込むときの方法をみてみましょう。  
読み込んだ文字列を `EditText` に表示する、という例です。

```
try{
    InputStream in = openFileInput( "a.txt" );
    BufferedReader reader =
        new BufferedReader(new InputStreamReader(in," UTF-8" ));
    String s;
    EditText et = (EditText)findViewById(R.id.edittext01);
    while((s = reader.readLine())!= null){
        et.append(s);
        et.append( "¥n" );
    }
    reader.close();
}catch(IOException e){
    e.printStackTrace();
}
```

---

ローカルファイルの読み込みは、`openFileInput()` メソッドを使います。  
引数にファイル名のみを指定して、入力ストリーム(`InputStream`)インスタンスを取得します。  
その `InputStream` を使って、`BufferedReader` インスタンスを生成します。  
その `BufferedReader` インスタンスの `readLine()` メソッドを使って、1 行ずつ読んでいきます。

---

## ● ローカルファイルの削除方法

ローカルファイルの削除方法を説明します。

```
-----  
deleteFile("a.txt");  
-----
```

上記のように、`deleteFile()`メソッドを使用してファイル名を指定するだけです。  
ファイルの削除が成功したか失敗したかは、戻り値によって返却されます。  
`true` ならば削除成功、`false` ならば削除失敗です。

サンプルコード)

```
package io.fileAccess;  
  
import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.io.OutputStream;  
import java.io.OutputStreamWriter;  
import java.io.PrintWriter;  
import android.app.Activity;  
import android.os.Bundle;  
import android.util.Log;  
import android.widget.TextView;  
  
public class FileAccessAndroid extends Activity {  
    // ファイル名  
    private static final String LOCAL_FILE = "data.txt";  
  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main);  
    }  
}
```

```

        TextView tv = (TextView) this.findViewById(R.id.TextView01);

        this.sampleFileOutput();// データをファイル書き込み保存
        tv.setText("ファイル処理 : " + this.sampleFileInput());// 読み込み、表示
    }

    /**
     * ファイル書き込み
     */
    private void sampleFileOutput() {
        OutputStream out;
        try {
            out = openFileOutput(LOCAL_FILE, MODE_PRIVATE | MODE_APPEND);
            OutputStreamWriter osw = new OutputStreamWriter(out, "UTF-8");
            PrintWriter writer = new PrintWriter(osw);

            // 追記する
            writer.append("あいうえお。");
            writer.append("かきくけこ。");

            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // sampleFileOutput()

    private String sampleFileInput() {
        InputStream in;
        String data;// メソッド返却データ
        String line;// ファイル 1 行データ格納用
        StringBuilder sb;// ファイル全データ格納用

        try {
            in = openFileInput(LOCAL_FILE);
            BufferedReader reader =

```

```

        new BufferedReader(new InputStreamReader(in, "UTF-8"));

        sb = new StringBuilder();
        // line に読み込み
        while ((line = reader.readLine()) != null) {
            sb.append(line);
            Log.d("FileAccess", line);
        }
        data = sb.toString();
    } catch (IOException e) {
        data = e.getMessage();
    }
    return data; // データを返却
} //sampleFileInput()
} //class_end

```

---

## ●SharedPreferences

キー項目と値のペアでデータを保存したい場合に利用します。

```
SharedPreferences.Editor editor = getSharedPreferences("favorite", 0);
```

### 保存できる値

- int
- float
- long
- boolean
- String

### 保存、読み込みのためのメソッド

- putInt(String key, int value)
- putFloat(String key, float value)
- putLong(String key, long value)
- putBoolean(String key, boolean value)

- putString(String key, String value)
- getInt(String key, int defValue)
- getFloat(String key, float defValue)
- getLong(String key, long defValue)
- getBoolean(String key, boolean defValue)
- getString(String key, String defValue)

;

以下の様な形式でキー値と値がペアで保存されます。

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string name="/pepin9876">This is my server name</string>
<string name="/akina-DJ">Akina Camel: Akina DJ</string>
<string name="/3121">Discotheque Saloon : GLAM SLAM</string>
<string name="/taiwanfm905">taiwanfm905</string>
<string name="/stream">hiroshima</string>
</map>
```

つぎに、clear()で消してみます。

**editor.clear().commit();**// 登録データを削除する

データは、以下の様な形になります。

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map />
```

サンプルコード)

```
import android.app.Activity;
import android.content.SharedPreferences;
import android.graphics.Color;
import android.os.Bundle;
```

```

import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;

//プリファレンスの読み書き
public class PreferencesEx extends Activity
    implements View.OnClickListener {
    private EditText editText;//エディットテキスト
    private Button    btnWrite;//書き込みボタン
    private Button    btnRead; //読み込みボタン

    //初期化
    @Override
    public void onCreate(Bundle icle) {
        super.onCreate(icle);
        requestWindowFeature(Window.FEATURE_NO_TITLE);

        //レイアウトの生成
        LinearLayout layout=new LinearLayout(this);
        layout.setBackgroundColor(Color.rgb(255,255,255));
        layout.setOrientation(LinearLayout.VERTICAL);
        setContentView(layout);

        //エディットテキストの生成
        editText=new EditText(this);
        editText.setText("",EditText.BufferType.NORMAL);
        setLLParams(editText,240,50);
        layout.addView(editText);

        //書き込みボタンの生成
        btnWrite=new Button(this);
        btnWrite.setText("書き込み");
        btnWrite.setOnClickListener(this);
        setLLParams(btnWrite);
    }

```

```

        layout.addView(btnWrite);

        //読み込みボタンの生成
        btnRead=new Button(this);
        btnRead.setText("読み込み");
        btnRead.setOnClickListener(this);
        setLLParams(btnRead);
        layout.addView(btnRead);
    }

    //ボタンクリックイベントの処理
    public void onClick(View v) {
        if (v==btnWrite) {
            //SharedPreferences オブジェクトの取得
            SharedPreferences pref=getSharedPreferences(
                "PreferencesEx",MODE_PRIVATE);

            //プリファレンスへの書き込み
            SharedPreferences.Editor editor=pref.edit();
            editor.putString("text",editText.getText().toString());
            editor.commit();
        } else if (v==btnRead) {
            //SharedPreferences オブジェクトの取得
            SharedPreferences pref=getSharedPreferences(
                "PreferencesEx",MODE_PRIVATE);

            //プリファレンスからの読み込み(3)
            editText.setText(pref.getString("text",""));
        }
    }

    //ライナーレイアウトのパラメータ指定
    private static void setLLParams(View view) {
        view.setLayoutParams(new LinearLayout.LayoutParams(
            LinearLayout.LayoutParams.WRAP_CONTENT,
            LinearLayout.LayoutParams.WRAP_CONTENT));
    }

```



```
}

//ライナーレイアウトのパラメータ指定
private static void setLLParams(View view,int w,int h) {
    view.setLayoutParams(new LinearLayout.LayoutParams(w,h));
}
}

} //class_end
```

---