

Android Parcelable を使ってクラスのメンバを一時保存

さて、前回のエントリで、Bundle で状態を保存する方法を書きました。
[Android Bundle で状態を保存](#)

ここでは、Bundle の Method (例えば putString と getString) を使ってパラメータを保存する方法を紹介しました。

しかーし、ここで問題が発生

「独自にデータクラスを用意していて、このクラスのメンバごと保存したいんだけど...」

さあ、この場合どうする？

ここで登場するのが **Parcelable** さんです。

Parcelable はリファレンス <http://developer.android.com/intl/ja/reference/android/os/Parcelable.html> に書いてあるように、Parcel にデータを書き／読みするためのインタフェースです。

"Interface for classes whose instances can be written to and restored from a Parcel. Classes implementing the Parcelable interface must also have a static field called CREATOR, which is an object implementing the Parcelable.Creator interface."

Parcelable インタフェースを実装したクラスは、Parcelable.Creator インタフェースを実装するオブジェクトである CREATOR と呼ばれる静的フィールドを持たなくてはなりません。

そして、ご丁寧に Parcelable インタフェースを実装するクラスの例が載ってます。

```
view plain print ?
01. public class MyParcelable implements Parcelable {
02.     private int mData;
03.
04.     public int describeContents() {
05.         return 0;
06.     }
07.
08.     public void writeToParcel(Parcel out, int flags) {
09.         out.writeInt(mData);
10.     }
11.
12.     public static final Parcelable.Creator<MyParcelable> CREATOR
13.         = new Parcelable.Creator<MyParcelable>() {
14.             public MyParcelable createFromParcel(Parcel in) {
15.                 return new MyParcelable(in);
16.             }
17.
18.             public MyParcelable[] newArray(int size) {
19.                 return new MyParcelable[size];
20.             }
21.         };
22.
23.     private MyParcelable(Parcel in) {
24.         mData = in.readInt();
25.     }
26. }
```

ここでは、Parcel に保存するパラメータとして、mData が1つですが、

例えば、名前、年齢、アドレスをメンバに持つクラス (ContactParcelable) を作るとこんな感じになります。

```
view plain print ?
01. public class ContactParcelable implements Parcelable {
02.     private String name;
03.     private int age;
04.     private String address;
05.
06.     public int describeContents() {
07.         return 0;
08.     }
09.
10.     public void writeToParcel(Parcel out, int flags) {
11.         out.writeString(name);
12.         out.writeInt(age);
13.         out.writeString(address);
14.     }
15.
16.     public static final Parcelable.Creator<ContactParcelable> CREATOR
17.         = new Parcelable.Creator<ContactParcelable>() {
18.             public ContactParcelable createFromParcel(Parcel in) {
19.                 return new ContactParcelable(in);
20.             }
21.
22.             public ContactParcelable[] newArray(int size) {
23.                 return new ContactParcelable[size];
24.             }
25.         };
26.
27.     private ContactParcelable(Parcel in) {
28.         name = in.readString();
29.         age = in.readInt();
30.         address = in.readString();
31.     }
32.
33.     public ContactParcelable(String name, int age, String address) {
34.         this.name = name;
35.         this.age = age;
36.         this.address = address;
37.     }
38. }
```

writeToParcel(Parcel out, int flags) で書き込む順番と ContactParcelable(Parcel in) で読み出す順番は同じにしなければなりません。

で、このクラスを onSaveInstanceState(Bundle) で保存し、onRestoreInstanceState(Bundle) で読み出すには、こんな感じでOK

```
view plain print ?
01. ContactParcelable contactParcelable;
02.
03. @Override
04. public void onCreate(Bundle savedInstanceState) {
05.     super.onCreate(savedInstanceState);
06.
07.     contactParcelable = new ContactParcelable("あんどろいど", 2, "android@gmail.com");
08. }
09.
10. @Override
11. protected void onSaveInstanceState(Bundle outState) {
12.     super.onSaveInstanceState(outState);
13.     /* ここで状態を保存 */
14.     outState.putParcelable("contact", contactParcelable);
15. }
16.
17. @Override
18. protected void onRestoreInstanceState(Bundle savedInstanceState) {
19.     super.onRestoreInstanceState(savedInstanceState);
20.     /* ここで保存した状態を読み出して設定 */
21.     contactParcelable = savedInstanceState.getParcelable("contact");
22. }
```

さらにさらに、もし、メンバがデータクラスのリストだった場合はどうなるか この場合も Parcelable を実装します。

ContactParcelable のリストをメンバに持つ AllContactParcelable を一時保存する場合

```
view plain print ?
01. public class AllContactParcelable implements Parcelable {
02.
03.     private ArrayList<ContactParcelable> contactList;
04.
05.     public int describeContents() {
06.         return 0;
07.     }
08.
09.     public void writeToParcel(Parcel out, int flags) {
10.         out.writeTypedList(contactList);
11.     }
12.
13.     public static final Parcelable.Creator<AllContactParcelable> CREATOR
14.         = new Parcelable.Creator<AllContactParcelable>() {
15.             public AllContactParcelable createFromParcel(Parcel in) {
16.                 return new AllContactParcelable(in);
17.             }
18.
19.             public AllContactParcelable[] newArray(int size) {
20.                 return new AllContactParcelable[size];
21.             }
22.         };
23.
24.     private AllContactParcelable(Parcel in) {
25.         contactList = in.createTypedArrayList(ContactParcelable.CREATOR);
26.     }
27.
28.     public AllContactParcelable(ArrayList<ContactParcelable> contactList) {
29.         this.contactList = contactList;
30.     }
31. }
```

out.writeTypedList(contactList); で書き込みし、contactList = in.createTypedArrayList(ContactParcelable.CREATOR); で読み出すのがポイント

onSaveInstanceState(Bundle) と onRestoreInstanceState(Bundle) での実装は同じ

参考ページ
・ [\[Android\] android.os.Parcelable / Parcel](#)

adakodaさんありがとうございます！

・ <http://developer.android.com/intl/ja/reference/android/os/Bundle.html>
・ <http://developer.android.com/intl/ja/reference/android/os/Bundle.html#putParcelable%28java.lang.String,%20android.os.Parcelable%29>
・ <http://developer.android.com/intl/ja/reference/android/os/Bundle.html#getParcelable%28java.lang.String%29>
⇒ ラベル: [Android](#)
投稿者 [yanzm](#) 時刻: [7:38](#)

自己紹介



uPhyca Inc. 代表取締役社長。Android のアプリつくったり、本書いたり、講演したりしてます。
GTUG Girls、droid girls マネージャー。
Google Developer Expert for Android
Suica Reader 作ってます。
「Master of Fragment」「わかる！ドメイン駆動設計」という本でした。

講演・執筆・講師の依頼は yanzm@uphyca.com までお願いします。

[詳細プロフィールを表示](#)

* ブログの記事(テキスト・画像)について



The contents in Y.A.Mの雑記帳 by yanzm is licensed under a [Creative Commons 表示 - 非営利 - 改変禁止 3.0 Unported License](#).

ツイート

@yanzmさんをフォロー

いいね！ 494人が「いいね！」しました。友達よりも先に「いいね！」しよう。



Y.A.Mの雑記帳を検索

検索

RSSに追加

投稿

コメント

ブログ アーカイブ

- ▶ 2020 (31)
- ▶ 2019 (30)
- ▶ 2018 (29)
- ▶ 2017 (44)
- ▶ 2016 (25)
- ▶ 2015 (27)
- ▶ 2014 (48)
- ▶ 2013 (73)
- ▶ 2012 (82)
- ▶ 2011 (129)
- ▼ 2010 (147)
 - ▶ 12月 (17)
 - ▶ 11月 (15)
 - ▶ 10月 (25)
 - ▶ 9月 (23)
 - ▶ 8月 (26)
 - ▶ 7月 (1)
 - ▶ 6月 (3)
 - ▶ 5月 (14)
 - ▶ 4月 (4)
 - ▼ 3月 (11)
 - [Android Ubuntu で adb devices ????? を直す](#)
 - [Android Parcelable を使ってクラスのメンバを一時保存](#)
 - [Android 特定の Intent \(Action\) を処理できる Activity \(アプリ\) の...](#)
 - [Android Bundle で状態を保存](#)
 - [Android 画面の縦横切り替え時に元の画面を保存](#)
 - [Amazon API レビュー情報を取得する](#)
 - [Androidアプリ Libraroid - 図書館予約 - 使い方](#)
 - [Androidアプリ Libraroid - 図書館予約 - アップデート情報](#)
 - [Android SearchManager ソフトキーボードを消すぜ！](#)
 - [Android ScrollBar \(ScrollView\) スクロールバーをカスタマイズ -](#)
 - [Android SearchManager 検索ボックスを使えぜ！](#)
 - ▶ 2月 (2)
 - ▶ 1月 (6)
- ▶ 2009 (148)
- ▶ 2008 (59)

ラベル

- ▶ [Android](#) (549)
- ▶ [JavaFX](#) (45)
- ▶ [Kotlin](#) (44)
- ▶ [日記](#) (31)
- ▶ [いろいろ](#) (26)
- ▶ [メモ](#) (19)
- ▶ [Material Design](#) (15)
- ▶ [Sublime Text 2](#) (15)
- ▶ [GWT](#) (14)
- ▶ [Google I/O 2013](#) (13)
- ▶ [pyROOT](#) (13)
- ▶ [python](#) (13)
- ▶ [TypeScript](#) (12)
- ▶ [本](#) (12)
- ▶ [記事](#) (12)
- ▶ [Android Wear](#) (9)
- ▶ [Google I/O](#) (9)
- ▶ [ListView](#) (8)
- ▶ [NFC](#) (8)
- ▶ [design](#) (8)
- ▶ [mockito](#) (8)
- ▶ [Dagger](#) (7)
- ▶ [Espresso](#) (7)
- ▶ [dart](#) (7)
- ▶ [AppEngine](#) (6)
- ▶ [HTML5](#) (6)
- ▶ [ML Kit](#) (6)
- ▶ [enchant MOON](#) (6)
- ▶ [イベント](#) (6)
- ▶ [Google I/O 2014](#) (5)
- ▶ [Java](#) (5)
- ▶ [TensorFlow Lite](#) (5)
- ▶ [XML](#) (5)
- ▶ [モバイルネイティブ](#) (5)
- ▶ [AIR](#) (4)
- ▶ [Amazon](#) (4)
- ▶ [Android Things](#) (4)
- ▶ [C2DM](#) (4)
- ▶ [Data Science](#) (4)
- ▶ [Eclipse](#) (4)
- ▶ [Firefox OS](#) (4)
- ▶ [Golang](#) (4)
- ▶ [Google Books Search API](#) (4)
- ▶ [Test](#) (4)
- ▶ [Ubuntu](#) (4)
- ▶ [polymer](#) (4)
- ▶ [volley](#) (4)
- ▶ [ADK](#) (3)
- ▶ [Android App](#) (3)
- ▶ [Frevo](#) (3)
- ▶ [Gimp](#) (3)
- ▶ [Google I/O 2012](#) (3)
- ▶ [Google I/O 2016](#) (3)
- ▶ [Google I/O 2017](#) (3)
- ▶ [Libraroid](#) (3)
- ▶ [つぶや貴](#) (3)
- ▶ [ドメイン駆動設計](#) (3)
- ▶ [Climbing](#) (2)
- ▶ [Google Chrome Extension](#) (2)
- ▶ [Kilimanjaro](#) (2)
- ▶ [Memo](#) (2)
- ▶ [linux](#) (2)
- ▶ [Actions on Google](#) (1)
- ▶ [Corona](#) (1)
- ▶ [Firebase](#) (1)
- ▶ [GDD](#) (1)
- ▶ [Geo](#) (1)
- ▶ [Google API](#) (1)
- ▶ [Google Assistant](#) (1)
- ▶ [Google TV](#) (1)
- ▶ [Google wave](#) (1)
- ▶ [Hanycomb](#) (1)
- ▶ [Hudson](#) (1)
- ▶ [ICS](#) (1)
- ▶ [Inkscape](#) (1)
- ▶ [Jelly Bean](#) (1)
- ▶ [Julia](#) (1)
- ▶ [Kinect](#) (1)
- ▶ [MQTT](#) (1)
- ▶ [ReVIEW](#) (1)
- ▶ [Slim3](#) (1)
- ▶ [Support Library](#) (1)
- ▶ [Truth](#) (1)
- ▶ [VirtualBox](#) (1)
- ▶ [adb](#) (1)
- ▶ [dar](#) (1)
- ▶ [device](#) (1)
- ▶ [ecology](#) (1)
- ▶ [jQuery Mobile](#) (1)
- ▶ [memo](#) (1)
- ▶ [uPhyca](#) (1)
- ▶ [ycard](#) (1)
- ▶ [イラスト](#) (1)
- ▶ [女子鑑](#) (1)

ページビューの合計



14,151,002

Blogger Syntax Highlighter

Syntax Highlighter for Code tag

0 件のコメント:

コメントを投稿

コメントを入力...

コメントの記入者: [okuda0715tech](#)

☐ お知らせを受け取る

[次の投稿](#)

[ホーム](#)

[前の投稿](#)

登録: [コメントの投稿 \(Atom\)](#)