ドコモメール連携アプリ開発ガイド
（新 IF 対応）

第 1.0.1 版

2015 年 12 月

改版履歴

版数	日付	記載箇所	内容	備考
1.0.0	2014 年 11 月	-	初版制定	
1.0.1	2015 年 12 月	1.5.	docomoID を d アカウントへ名称変更	

目次

1. はじめに

ドコモメールと外部アプリ間でメールデータの受け渡しを簡易に行える連携インタフェースを提供します。

連携アプリのご提供にあたっては以下の内容を遵守いただけますようお願いいたします。

1. 本 IF に対応した連携アプリを提供することにより、ドコモメールアプリからデータを取得するにあたっては、事前にデータの利用目的を明示したうえでユーザの許諾を得ることとし、当該利用目的以外の目的でデータを利用しないこと。また、ドコモメールアプリから取得したデータを端末外に送信する場合には、事前に送信先を明示したうえで当該外部送信についてもユーザの許諾を得ること。
2. 本 IF に対応した連携アプリの提供にあたり、海外在圏時に通信を行う仕様とする場合には、その通信を許可するかどうかの設定を設けるなどして事前にユーザの許諾を得ること。
3. 本仕様書を第三者に開示する場合には、当該第三者をして上記 1,2 に記載の事項を遵守させること。

1.1. 本資料について

本資料ではドコモメールと連携する外部アプリ(以降、「連携アプリ」と呼びます)を開発する方のためにアプリ間の連携インタフェースを説明します。本資料は Android アプリ開発の知識がある方を対象としています。

また、本資料内での「連携インタフェース」は新 IF のことを指します。

1.2. 参照文書

参照文書を以下に示します。

[1] ドコモメール／sp モードメール連携アプリ作成ガイド

1.3. 用語説明

項番	用語	説明
1	アプリデータ領域	/data/data/”アプリのパッケージ名” 配下の領域のこと
2	グローバルアクセス領域	SD カード領域や内蔵ストレージ領域など、第三者アプリもアクセス可能な領域のこと

1.4. サービス概要

アプリ間連携のイメージを以下の図 1.4-1 に示します。

図 1.4-1 サービス概要

1.5. 対応 OS・対応バージョン

・Android OS 4.0 以上

・スマートフォン向け UI のドコモメールの場合、以下のバージョンより対応

バージョン 60400 （d アカウント設定対応機種向け）

バージョン 40400 （d アカウント設定非対応機種向け）

・タブレット向け UI のドコモメールの場合、以下のバージョンより対応

バージョン 60600 （d アカウント設定対応機種向け）

バージョン 40600 （d アカウント設定非対応機種向け）

・SP モードメールは非対応

2. 連携インタフェース概要

連携インタフェースには以下の 2 種類の起動方法があります。

・ドコモメールから連携アプリ起動する

・連携アプリからドコモメール起動する

いずれの起動も Intent を利用したデータ共有方法です。

2.1. ドコモメールから連携アプリ起動

ドコモメールから連携アプリを起動する流れは以下の図 2.1-1 のとおりです。

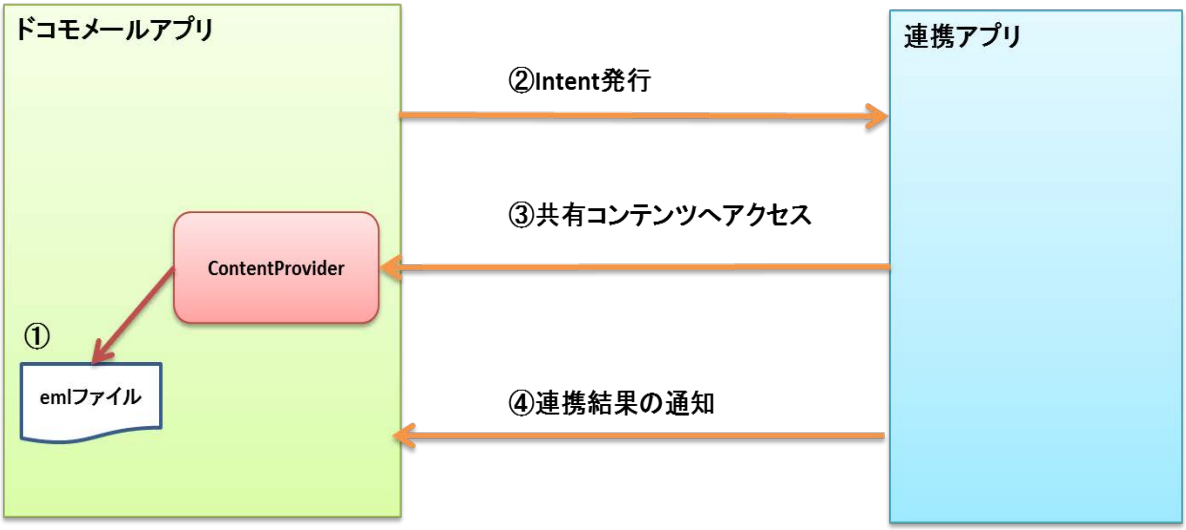

図 2.1-1　ドコモメールから連携アプリ起動

① 編集中のメールのヘッダ、本文、添付データ情報を持つ共有ファイルをアプリデータ領域に作成します。

② 明示的 Intent を発行し、連携アプリを起動します。Intent には共有ファイルにアクセスするための URI を設定します。Intent に設定するデータの詳細については 3.1.1 を参照してください。

③ ②で設定された URI を使用して ContentProvider 経由で共有ファイルにアクセスし、共有ファイルの編集を行います。※1

④ 連携結果をドコモメールに通知します。連携結果の Intent に設定するデータの詳細については 3.1.2 を参照してください。連携結果の反映後、ドコモメールが作成した共有ファイルは削除されます。宛先情報が付与されている場合は無視されます。

※1 ドコモメールの共有ファイルを直接編集する以外に編集結果のファイルをグローバルアクセス領域や連携アプリのアプリデータ領域に保存し、それを編集結果として通知することもできます。
しかし、以下のような問題があるため、ドコモメールから連携アプリが起動された場合、ドコモメールの共有ファイルを直接編集する方法を推奨します。
・グローバルアクセス領域を利用する場合、悪意あるアプリケーションが編集結果にアクセスできてしまう。
・連携アプリのアプリデータ領域を利用する場合、連携結果ファイルの削除契機を連携アプリで判断する必要がある。

また、ドコモメールの共有ファイルには 3.1.1 に記載の通り FLAG_GRANT_READ_URI_PERMISSION/ FLAG_GRANT_WRITE_URI_PERMISSION を利用した一時的なアクセス権限が付与されております。このアクセス権限はフレームワーク側により自動的に剥奪されてしまうため、恒久的なアクセス権限ではないことを意識した実装としてください。

なお、連携アプリが悪意のあるアプリケーションから起動されると、本来と異なる目的で連携アプリが利用されるリスクが想定されます。このため、参照文書[1]に記載されている「証明書ハッシュ値」を用いた呼び出し元アプリの判定を行うことを推奨します。

2.2. 連携アプリからドコモメール起動

連携アプリからドコモメールを起動する流れは以下の図 2.2-1 のとおりです。

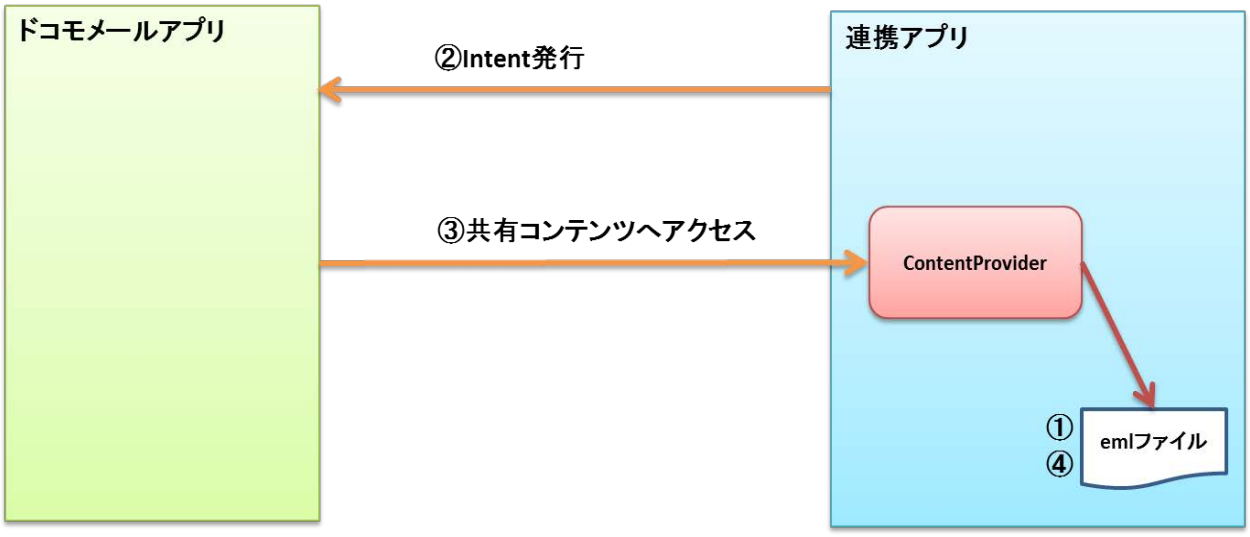

図 2.2-1 連携アプリからドコモメール起動

① 連携アプリ側で編集したヘッダ、本文、添付データ情報を持つ共有ファイルをアプリデータ領域に作成します。

② Intent を発行し、ドコモメールを起動します。Intent には共有ファイルにアクセスするための URI を設定します。Intent に設定するデータの詳細については 3.2 を参照してください。

③ ②で設定された URI を使用して ContentProvider 経由で共有ファイルにアクセスし、メール作成画面に編集したメールの情報を反映します。宛先情報が付与されている場合は無視されます。

④ 連携終了後、共有ファイルを削除します。共有ファイル削除の詳細については 4.5 を参照してください。

3. Intent 仕様

Intent の仕様について記載します。

3.1. ドコモメールから連携アプリ起動

3.1.1. ドコモメールから連携アプリ起動

ドコモメール発の Intent ではドコモメールから連携アプリに対して明示的 Intent を発行します。
Intent パラメータは以下の表 3.1-1 の通りです。
連携起動に対応しているアプリかどうかをチェックする際に、この Intent 情報が使用されます。

表 3.1-1　ドコモメールから連携アプリに渡される Intent 情報

項番	要素	型	値	説明
1	action	String	jp.co.nttdocomo.carriermail.v2.ACTION_EDIT	-
2	category	String	Intent.CATEGORY_DEFAULT	-
3	data	Uri	ドコモメールが作成した共有ファイルにアクセスするための URI	例: content://provider_auth/hoge/hoge.eml
4	type	String	message/rfc822	ファイル Type を指定する。 文字列は完全一致とする。
5	flags	int	Intent.FLAG_GRANT_READ_URI_PERMISSION と Intent.FLAG_GRANT_WRITE_URI_PERMISSION	共有ファイルの URI にアクセスするための権限。

3.1.2. ドコモメールに連携結果の通知

連携アプリではドコモメールとの連携が終了後、android.app.Activity クラスの setResult()でドコモメールに対して結果情報を返却してください。
連携結果情報に指定する resultCode は以下の表 3.1-2 の通りです。

表 3.1-2　連携結果情報に指定する resultCode

項番	要素	型	値	説明
1	-	int	Activity.RESULT_OK	連携リクエストが正常に処理されたことを示す。
2	-	int	Activity.RESULT_CANCEL	連携リクエストの処理に失敗、または拒否されたことを示す。

また、連携結果情報に指定する Intent パラメータは以下の表 3.1-3 のとおりです。

表 3.1-3　ドコモメールに渡す連携結果の Intent 情報

項番	要素	型	値	説明
1	action	String	設定不要	-
2	category	String	設定不要	-
3	data	Uri	編集結果にアクセスするための URI 【ドコモメールの共有ファイルを編集した場合】（推奨） ドコモメールから渡された URI 【連携アプリのアプリデータ領域に編集結果を作成し、ドコモメールに渡す場合】（非推奨） 連携結果ファイルにアクセスするための URI	例: content://provider_auth/hoge/hoge.eml(ドコモメールから渡された URI)（推奨） または [連携結果ファイルにアクセスするための連携アプリの URI(content://~)]（非推奨） [連携結果ファイルにアクセスするためのファイルパス(file://~)] （非推奨）
4	type	String	message/rfc822	ファイル Type を指定する。 文字列は完全一致とする。
5	flags	int	Intent.FLAG_GRANT_READ_URI_PERMISSION	共有ファイル URI にアクセスするための権限 連携アプリのアプリデータ領域に連携結果を保存した場合のみ設定

3.2. 連携アプリからドコモメール起動

連携アプリ発の Intent では連携アプリからドコモメールに対してドコモメールのパッケージ名と Action 名を指定して Intent を発行します。

連携アプリは android.app.Activity.startActivity ()を使用します。

Intent パラメータは以下の表 3.2-1 の通りです。

表 3.2-1　連携アプリからドコモメールに渡される Intent 情報

項番	要素	値	値	説明
1	action	String	jp.co.nttdocomo.carriermail.v2.ACTION_EDIT	-
2	category	String	Intent.CATEGORY_DEFAULT	-
3	data	Uri	連携アプリが作成した共有ファイルにアクセスするための URI	例: content://provider_auth/hoge/hoge.eml URI の scheme が「content」でない場合、ドコモメールは起動することができません。「file」scheme を利用されたい場合は参照文書[1]の連携方法を利用してください。 （ただし、「file」scheme の利用は非推奨となります）
4	type	String	message/rfc822	ファイル Type を指定する。 文字列は完全一致とする。
5	flags	int	Intent.FLAG_GRANT_READ_URI_PERMISSION	ContentProvider にアクセスするための権限
6	package	String	jp.co.nttdocomo.carriermail	ドコモメールのパッケージ名

4. 共有ファイル仕様

ドコモメールと連携アプリ間で共有するファイルの仕様について記載します。

4.1. ファイル形式

ドコモメールと連携アプリ間で受け渡すファイルは eml ファイルとします。

4.2. ファイル共有領域

ドコモメールと連携アプリ間でやり取りされる eml ファイルはファイル共有領域に格納されます。
ファイル共有領域は特別な理由がない限りはアプリデータ領域を利用することを推奨します。
ファイル共有領域としてグローバルアクセス領域を利用することは非推奨です。グローバルアクセス領域に eml ファイルを保存する際、連携アプリ以外のアプリが Read/Write できる権限を指定する必要があります。ドコモメールの共有ファイルを編集して、同 URI のアプリデータ領域に eml ファイルを保存する際、権限の設定は不要です。
OS が Android4.4 以降の端末の場合、外部 SD への書き込みが制限されます。

4.3. ファイルサイズ

eml ファイル生成前のメールサイズはドコモメールで扱える上限の 10MB 未満としてください。
・ドコモメールアプリは送信時にサイズチェックを行い、10MB 超過の場合は送信しません。
このため上限 10MB のデータを連携すると、その他情報付加により送信できないことを留意してドコモメールへデータを連携してください。
・インライン画像が 20 種類存在する場合、20 種類を超過する画像は削除します。
・インライン画像が合計 2MB 以上の場合、2MB を超える画像は削除します。
・本文は HTML 文を含めて 500KB までを有効とし、500KB を超える部分は削除します。
HTML 文が途中で削除される場合、その削除された HTML 文は残った範囲でテキスト表示となります。

4.4. ファイル生成時の注意

eml ファイル生成時は ANR 等を考慮した設計を行ってください。

処理時間が長くなる場合は処理中である旨を表示するとともに、ユーザが中断できるようにしてください。

4.5. ファイル削除

連携アプリからドコモメール起動の場合、適切な削除タイミングを連携アプリ側が判断し、連携アプリが作成した eml ファイルを削除してください。
startActivity()呼び出しでドコモメールを起動するため、連携アプリは連携完了通知を受け取ることができません。
また、過去に連携アプリが作成した eml ファイルが存在する場合も eml ファイルを削除してください。
ドコモメールから連携アプリ起動の場合、ドコモメールが作成した eml ファイルは連携アプリ側で削除する必要はありません。しかし、連携アプリ側で連携結果を作成し、ドコモメールに通知する場合、適切な削除タイミングを連携アプリ側が判断し、連携結果ファイルを削除する必要があります。

4.6. eml ファイルの MIME マルチパート解析

MIME マルチパートの解析は連携アプリ側にて対応してください。

4.7. 対応 HTML タグ

ドコモメールが対応する HTML タグを使用してください。
対応する HTML タグは以下の URL を参照してください。
http://www.nttdocomo.co.jp/service/developer/make/content/deco_mail/tag/index.html
ドコモメールが対応していない HTML タグを eml ファイルに記載して、ドコモメールに渡した場合、非対応の HTML タグが削除されてドコモメールで表示されます。

5. その他要件

5.1. ドコモメール起動可否チェック

連携アプリからのドコモメール起動では、必ず以下の方法でドコモメールが連携インタフェースに対応しているかどうかを確認してください。

【連携インタフェース対応チェック方法】
android.content.pm.PackageManager クラスの queryIntentActivities(Intent intent, int flags)を使用して表 3.2-1 の Intent 情報に対応したアプリのパッケージ情報を取得します。チェックに必要な Intent の情報は表 5.1-1 に示します。
パッケージ情報が取得できればドコモメールは連携インタフェースに対応しています。

表 5.1-1　ドコモメール起動可否チェックで必要な Intent 情報

項番	要素	値	値	説明
1	action	String	jp.co.nttdocomo.carriermail.v2.ACTION_EDIT	固定値
2	category	String	Intent.CATEGORY_DEFAULT	固定値
3	data	Uri	[例] content://check/dummy.eml	・“content”スキームであること。 ・”.eml”拡張子であること
4	type	String	message/rfc822	固定値
5	package	String	jp.co.nttdocomo.carriermail	固定値

6. サンプルコード

6.1. ドコモメールから連携起動された場合

ドコモメールから連携アプリが起動された場合の連携アプリ側のサンプルコードを記載します。
サンプルではドコモメールから受け取った URI からメール情報を読み取り、同 URI にメール情報を書き込んで連携結果を通知するという処理を行っています。

6.1.1. ドコモメールからの連携を受ける

ドコモメールからの連携起動に対応するために Intent-Filter を設定します。

AndroidManifest.xml

```
<activity
    android:name="com.example.hoge.HogeActivity"
    android:label="@string/hogeactivity_name" >
    <!--  ドコモメールからの連携起動を受けるための Intent-Fileter の設定  -->
    <intent-filter>
        <action android:name="jp.co.nttdocomo.carriermail.v2.ACTION_EDIT" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:mimeType="message/rfc822" />
    </intent-filter>
</activity>
```

6.1.2.共有ファイルにアクセスする

ドコモメールから渡された URI 情報を使用して ContentProvider 経由でドコモメールが作成した共有ファイルの情報を読み込みます。

HogeActivity.java

```
//ドコモメールから渡された Intent を取得
Intent intent = getIntent();
//Intent から共有ファイルにアクセスするための URI の取得
Uri uri = intent.getData();

InputStream in = null;
OutputStream out = null;
try {
    //共有ファイルの IntputStream を取得する
    in = getContentResolver().openInputStream(uri);
    //共有ファイルの文字コードは UTF-8
    InputStreamReader reader = new InputStreamReader(in, "UTF-8");
    StringBuilder sb = new StringBuilder();
    char[] buf = new char[1024];
    int numRead;
    while (0 <= (numRead = reader.read(buf))) {
        sb.append(buf, 0, numRead);
    }

    //共有ファイルの中身を読み込み
    String mailData = sb.toString();

    //共有ファイルの中身を解析
    //本文部分に編集を加える

    //共有ファイルの OutputStream を取得する
    out = getContentResolver().openOutputStream(uri);
    //編集したものを書き込む
    byte[] buffer = mailData.getBytes();
    out.write(buffer);
    out.flush();
```

```
} catch (Exception e) {
    //エラー発生時の処理
} finally {
    //取得した InputStream、OutputStream の close 処理
}
```

6.1.3. ドコモメールへ連携結果通知する

編集終了後、ドコモメールへ連携結果通知を行います。

URI には編集結果にアクセスするための情報を設定します。

・ドコモメールの共有ファイルを直接編集した場合、渡された URI をそのまま設定します。

HogeActivity.java

```
Intent intent = new Intent();

//URI,type の設定
intent.setDataAndType(uri, "message/rfc822");

if (連携成功を通知する場合) {
    setResult(RESULT_OK, intent);
} else {
    //連携失敗を通知する場合
    setResult(RESULT_CANCEL, intent);
}

finish();
```

6.2. 連携アプリからのドコモメール起動する場合

連携アプリからドコモメールを起動する場合の連携アプリ側のサンプルコードを記載します。

6.2.1. 共有ファイルにアクセスする ContentProvider の定義

サンプル例では一時許可 ContentProvider の定義例を記載しています。

AndroidManifest.xml

```
<!-- 共有ファイルにアクセスするために使用する ContentProvider の定義 -->
<provider
    android:name="HogeProvider"
    android:authorities="com.example.hoge.hogeprovider"
    android:grantUriPermissions="true"
    android:exported="false">
</provider>
```

6.2.2. ドコモメールを起動する

ドコモメールを起動するために Intent 設定を行います。URI には連携アプリのアプリデータ領域に保存した共有ファイルにアクセスするための情報(content://～)を設定してください。ドコモメール側は渡された URI を使用して連携アプリの ContentProvider 経由で連携アプリが作成した共有ファイルにアクセスします。なお、共有ファイルの削除については 4.5 を参照してください。

```
Intent intent = new Intent();

//アプリデータ領域に共有ファイルを作成していることが前提
//共有ファイルにアクセスするための URI を作成する
Uri uri = Uri.parse("content://com.example.hoge.hogeprovider/～")

//ContentProvider へのアクセス権限(読み込み可能)の設定
intent.setFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);

//category の設定
intent.addCategory(Intent.CATEGORY_DEFAULT);

//URI,type の設定
intent.setDataAndType(uri, "message/rfc822");

//action の設定
intent.setAction("jp.co.nttdocomo.carriermail.v2.ACTION_EDIT");

//ドコモメールのパッケージの設定
intent.setPackage("jp.co.nttdocomo.carriermail");

startActivity(intent);
```

6.2.3. ドコモメール起動可否チェック

ドコモメール起動可否チェックの例を示します。

```
Intent intent = new Intent();

//URI,type の設定
Uri uri = Uri.parse("content://check/dummy.eml")
intent.setDataAndType(uri, "message/rfc822");

//action の設定
intent.setAction("jp.co.nttdocomo.carriermail.v2.ACTION_EDIT");

//ドコモメールのパッケージの設定
intent.setPackage("jp.co.nttdocomo.carriermail");

//category の設定
intent.addCategory(Intent.CATEGORY_DEFAULT);

//ドコモメールが連携インタフェースに対応しているかどうかのチェック
PackageManager packageManager = getApplicationContext().getPackageManager();
List<ResolveInfo> ver2Applications = packageManager.queryIntentActivities(intent, 0);
if (0 < ver2Applications.size()) {
    //対応している場合の動作
} else {
    //対応していない場合の動作
}
```