

## 「Build Your First Mobile Flex Application」の日本語訳

本ドキュメントは、[rictus.com/muchado/](http://rictus.com/muchado/)で公開されている記事「Build Your First Mobile Flex Application」をヒム・カンパニー 永井勝則が自主的に日本語に訳したものです。

<http://www.himco.jp/>

[knagai@himco.jp](mailto:knagai@himco.jp)

(2010/11)

本ドキュメントの原文は

<http://www.rictus.com/muchado/2010/10/27/tutorial-from-build-your-first-mobile-flex-application-lab/>

ページ内のリンクからダウンロードできます。

## 初めてのモバイル Flex アプリケーションの作成

Narciso Jaramillo

2010/11

### はじめに

Flash Builder “Burrito”と Flex SDK “Hero”の新しいモバイル開発のプレビューリリースへようこそ！これはプレビューリリースなので、既知のバグやパフォーマンス、使い勝手の問題や欠落している機能があります。

以降に示すこの囲み内の項目はオプションの情報で、さらに理解を深めたいときの参考になりますが、次の段階に進むために必ず読む必要のある内容ではありません。

このチュートリアルを進めるには TwitterTrendsAssets.zip ファイルで提供しているアセットを使用する必要があります。このファイルは Adobe Labs の Sample Applications and Tutorials ページからダウンロードできます ([http://labs.adobe.com/technologies/flexsdk\\_hero/samples/](http://labs.adobe.com/technologies/flexsdk_hero/samples/))。

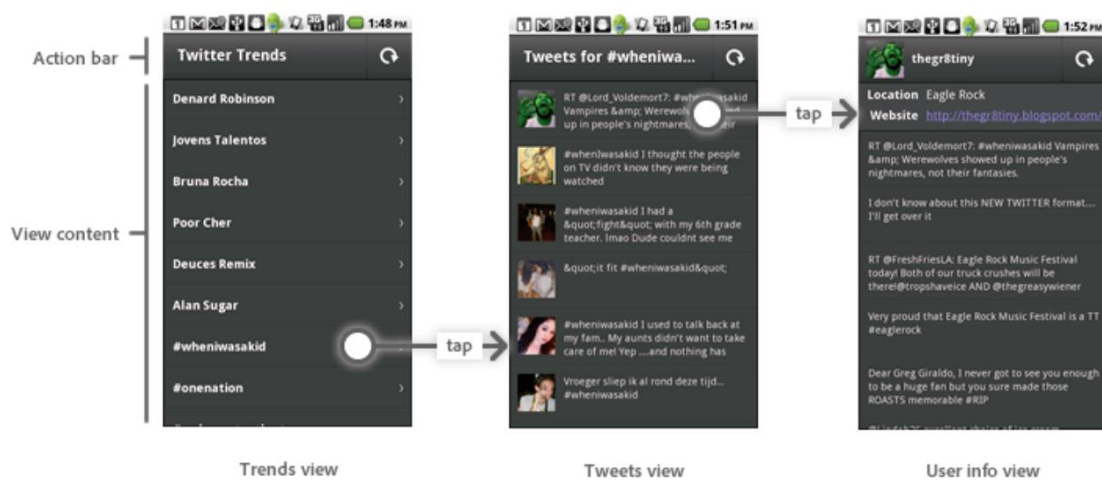
演習はアプリケーションを作成する手順に沿って進みますが、TwitterTrendsAssets.zip ファイルには演習の各段階から開始できる FXP プロジェクトファイルが含まれているので、演習の途中でよく分からなくなった場合でも、その演習の初めからやり直したり、次の演習の最初まで飛ばすこともできます。静的なデータ（変化しないデータ）を使用する場合には staticCheckpoints フォルダ内の FXP ファイルを使用し、ライブデータ（変化する生のデータ）を使う場合には liveCheckpoints フォルダ内の FXP ファイルを使ってください（静的なデータとライブデータについては演習3で見ていきます）。

FXP ファイルをインポートするには次のようにします。

1. (オプション) 既存の TwitterTrends プロジェクトがある場合には、[パッケージエクスプローラー]でプロジェクトのルートをクリックし[削除]を選びます。これによってこれから読み込む新しいバージョンと見間違えることがなくなります。
2. [ファイル]→[読み込み]を選択します。
3. [Flash Builder]フォルダを展開し、[Flash Builder プロジェクト]を選択します。
4. [次へ]をクリックします。
5. [ファイル]フィールドの右にある[参照]ボタンをクリックし、目的の FXP ファイルを指定します。[OK]をクリックします。
6. [終了]をクリックします。

## 作成するアプリケーションについて

このチュートリアルでは簡単なモバイル用 Twitter ブラウザを作成していきます。このアプリケーションでは Twitter のトレンドトピックのリストが表示され、ユーザーは、トピックをタップすることでそのトピックのツイートリストを見ることができ、ツイートをタップすることでそのツイートを投稿したユーザーのさらに詳しい情報を見ることができます。ユーザーはデバイスの戻るボタンを使って1つ前のビューに移動でき、縦横の画面表示を切り替えることもできます。



モバイル Flex アプリケーションはデフォルトで、一連のビューで組み立てられます。ビューとはユーザーが行き来できる個々の UI スクリーンのことです。上図のビューには、現在のビューのタイトルを含むアクションバーや、ナビゲーション、更新などのコントロールが置かれています。

このサンプルの最初のビューである Trends ビューには Flex の List コンポーネントが1つ含まれています。モバイルアプリケーションの List コンポーネントは、タッチ入力にตอบสนองしてスクロールします。このビューのアクションバーはビューのタイトル(Twitter Trends)と更新ボタンを表示しています。ほかのビューも同様な構造をしていますが、3つめの User Info ビューのアクションバーは、選択したユーザーの写真を表示するようにカスタマイズされています。

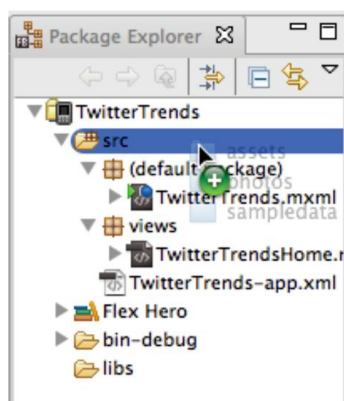
## 演習1: プロジェクトの作成とアクションバーの設定

この演習では、Flash Builder でモバイルプロジェクトを作成し、デザインモードを使ってアプリケーションのグローバルな(どの画面でも表示される)アクションバーを設定します。

Flex モバイルプロジェクトを作成し、プロジェクトのアセットを設定する

1. Flash Builder を起動します。
2. [ファイル]→[新規]→[Flex モバイルプロジェクト]を選びます。
3. [プロジェクト名]に TwitterTrends と入力し[次へ]をクリックします。
4. [モバイル設定]で、以下を確認します(いずれもデフォルトの設定です)。
  - (ア) [Google Android]が選択されている
  - (イ) [モバイルアプリケーション]が選択されている
  - (ウ) [自動的に方向を変更]が選択され、[フルスクリーン]は選択されていない
5. [終了]をクリックします。
6. このチュートリアルで提供している TwitterTrendsAssets.zip ファイルを展開します。
7. assets フォルダと photos、sampledata フォルダを、Flash Builder の左側にあるパッケージエクスプローラーの src フォルダにドラッグします。

Note: 静的なサンプルデータでなく、ライブの Twitter API を使用したい場合には、assets フォルダだけをドラッグします。photos フォルダと sampledata フォルダは必要ありません(とはいえ、これらをドラッグしてもアプリケーションが大きくなるだけで、実害はありません)。静的なデータとライブデータについては演習3を参照してください。



Flash Builder は2つの MXML コンポーネントを生成します。1つは TwitterTrends.mxml で、これはメインアプリケーションファイルです。もう1つは \_TwitterTrends.mxml で、これは View にもとづく、アプリケ

ーションの最初のビューです。

訳者注:

\_TwitterTrends.mxml は原文と図では TwitterTrendsHome.mxml になっていますが、訳者の作業では \_TwitterTrends.mxml が自動的に作成されました。

デスクトップの Flex と異なり、メインアプリケーションには通常、多くの UI コンテンツは配置せず、ほとんどの UI は個々のビューに置きます。唯一の例外は次に見ていくアクションバーです。

グローバルなコンテンツをメインアプリケーションファイルのアクションバーに追加する

アクションバーはモバイル Flex アプリケーションの特殊なコンポーネントで、個々のビューではなく、アプリケーション全体の一部を構成します。その項目はメインの MobileApplication でも個々のビューでも追加できます。このアプリケーションではすべてのビューで利用できる更新ボタン (refreshBtn) を追加します。後の演習9では、アクションバーを特定のビューでカスタマイズする方法を見ていきます。

1. エディタの表示をメインアプリケーションファイルの TwitterTrends.mxml に切り替えます。
2. 空の <fx:Declarations> ... </fx:Declarations> タグを消去します。
3. <s:MobileApplication> と </s:MobileApplication> タグの間に次のコードを記述します。

```
<s:actionContent>
    <s:Button id="refreshBtn"
              icon="@Embed(source='assets/refresh48x48.png')"
              click="Object(navigator.activeView).refresh()"/>
</s:actionContent>
```

アクションバーには、左の navigationContent と中央の titleContent、右の actionContent という3つの領域があります。アクションバーのタイトル領域にはデフォルトで、各ビューのタイトルに関連づけられたテキストコンポーネントが含まれるので、わざわざ指定する必要はありません。

このチュートリアルでは、アクションバーのコンテンツはメインアプリケーションでだけ設定しますが、コンテンツはそのビュー特定のコンテンツを表示したい場合など、個々のビューの領域でも設定できます。個々のビューでコンテンツ領域のどれかを設定すると、アプリケーションはそのコンテンツを優先します。

ここではグローバルなアクションバーのコンテンツに更新ボタンを追加したので、ボタンのアクションは現在表示されている、どれかが特定できないいずれかのビューに依存することになります。したがって click ハンドラ

では現在アクティブなビューを取得し、その `refresh()`メソッドを呼び出しています。このメソッドはこの後作成する各ビューで実装します。

## 演習2: Trends ビューのレイアウト

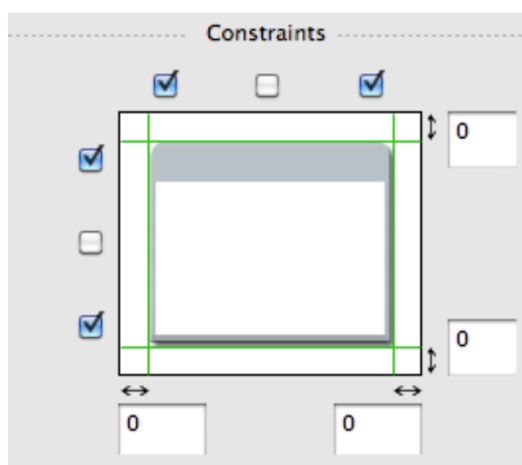
この演習ではデザインモードを使って、最初のビュー、\_TwitterTrends.mxml のコンテンツをレイアウトします。このビューは Twitter からのトレンドトピックのリストを表示します。またレイアウトの異なる向きと画面サイズの変更機能を試します。

### ビューのレイアウト

1. \_TwitterTrends.mxml ファイルに切り替えます。
2. [デザイン]ボタンをクリックしデザインモードに切り替えます。

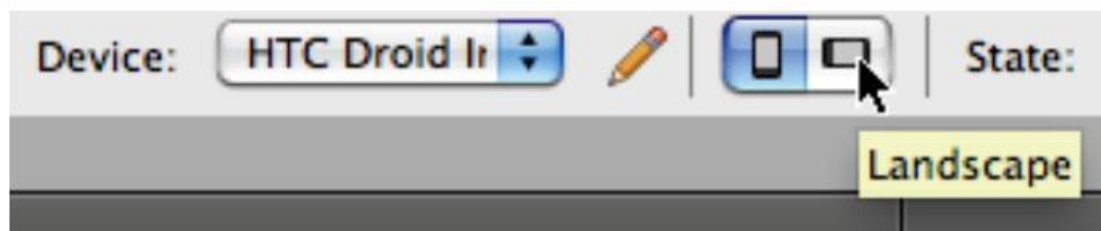


4. ビューで更新ボタン(右回りに回転する矢印アイコン)が壊れた画像アイコンとして表示される場合には、[デザイン]ボタンの右にある緑の[更新]アイコンをクリックします。
5. 右にある[プロパティ]パネルで、[タイトル]を Twitter Trends に設定します。
6. 左の[コンポーネント]パネルから List コントロールを、アクションバーの下のメインのビュー領域にドラッグします。
7. リストは View の左端に正確に吸着させ、その上端がアクションバーのすぐ下に来るように配置します。
8. リストが View の右端と下端まで全体を埋めるように、リストのサイズを変更します。そのためにはビューをスクロールする必要があるかもしれません。
9. [プロパティ]パネルを[制約]までスクロールダウンします。上下左右のチェックボックスを選択し、List を View の端に固定させます。ボックス内の値が 0 でない場合には、下図のように 0 に変更します。



## ビューのサイズ変更機能のテスト

1. エディタの \_TwitterTrends.mxml のタブをダブルクリックし、Flash Builder のほかのパネルを隠します。
2. デザインビューのツールバーで、[横方向]ボタンをクリックします。するとアクションバーと List のサイズが新しい向きに合わせて適切にリサイズされます。



3. [デバイス]ドロップダウンメニューで、たとえば Motorola Droid など異なる画面サイズのデバイスを選択します。するとビューは水平方向に大きくなり、ビューのすべての要素が適切にリサイズされます。
4. [縦方向]ボタンをクリックします。エディタのタブを再度ダブルクリックし、パネルを元に戻します。

### 訳者注:

この[横方向]と[縦方向]ボタンは、訳者の Flash Builder では、上図のような”立派なボタン”ではなく、1つの赤い矩形で表示されました。矩形の右端と左端をクリックすると、方向が切り替わるようです。プレビューリリースのバグと言うより言い様がありません。

### 演習3: Trends ビューをデータに接続させる

この演習では、Twitter からのトレンドトピックを Trends ビューの List に入れます。

Note: このチュートリアルを進めるに当たっては、選択肢が2つあります。Twitter API からの実際のライブデータを使用することもできますし、このチュートリアルで提供している静的なデータを使用することもできます。ただしライブ API を使用するときには、Twitter 側がそのサービスにアクセスされる回数を任意の期間に制限している可能性のあることに注意が必要です。MAX lab ではこれを避けるため、われわれは静的なデータを使用します。とは言えこのチュートリアルを自力でやってみようと思われる方は、ライブデータサービスに挑戦してみてください。

静的なデータ	ライブデータ
<ol style="list-style-type: none"><li>1. [データ]→[XML に接続]を選びます。</li><li>2. [パス]フィールドの右にある[参照]ボタンをクリックし、src/sampleddata フォルダにある trends.xml ファイルを選択します。</li><li>3. [サービス名]を TrendsService に設定します。後はそのまま変更せず[終了]をクリックします。</li><li>4. 下部にある[データとサービス]パネルに getData()が表示されるので、右クリックして表示されるコンテキストメニューから[サービスの呼び出しを生成]を選択します。すると[ソース]ビューに切り替わり、新たに生成された getData()関数が表示されます。</li><li>5. コード内の getData() 関数の名前を getTrends()に変更します(この関数はサービスの getData()を呼び出します)。</li></ol> <pre>protected function getTrends() {     getDataResult.token =         trendsService.getData(); }</pre>	<ol style="list-style-type: none"><li>1. [データ]→[HTTP に接続]を選びます。</li><li>2. [名前]に入力されている Operation1 を getTrends に変更します。</li><li>3. [URL]を http://search.twitter.com/trends.json に変更します。</li><li>4. [サービス名]に TwitterService と入力し、[終了]をクリックします。</li><li>5. 下部にある[データとサービス]パネルに getTrends()が表示されるので、右クリックして表示されるコンテキストメニューから[戻り値の型を設定]を選択します。</li><li>6. [サンプルデータから戻り値の型を自動検出]が選ばれていることを確認し、[次へ]をクリックします。</li><li>7. 設定が必要なパラメータはないので、[次へ]をクリックします。</li><li>8. [作成する新規データ型の名前を入力]で Trends を入力します。</li><li>9. 混乱しないように、trends プロパティの型を Trends[]から Trend[]に変更します。これによって各 trend オブジェクトが Trends ではなく Trend と呼べるようになります。</li><li>10. [終了]をクリックします。</li><li>11. getTrends()のコンテキストメニューから[サー</li></ol>

	ビスの呼び出しを生成]を選択します。すると[ソース]ビューに切り替わり、新たに生成された <code>getTrends()</code> 関数が表示されます。
--	--

コードの<Declarations>タグには、サービスオブジェクト(TwitterService か TrendsService。使用するデータによって変わります)と CallResponder という2つの項目が追加されています。サービスオブジェクトはサービスへのネットワーク呼び出しを初期化するメソッドを持っています。CallResponder はサービス呼び出しを監視し、それが完了したら、サービス呼び出しからデータが返されたことをほかのコンポーネントに通知するイベントを送出します。

静的なデータを使用するときには、個々のサービスが、データを返す単一の `getData()`メソッドを使って各 XML ファイルにアクセスします。HTTP サービスを使用するときには、同じサービスに異なる URL 用の複数のメソッドを作成することもできます。

ユーザーがナビゲートしたかビューを更新したときにサービスを呼び出す

メインの TwitterTrends.mxml アプリケーションファイルのアクションバーには、各ビューで `refresh()`メソッドを呼び出せる更新ボタンを置いたことを思い出してください。このメソッドは `_TwitterTrends.mxml` 内で、適切なサービス呼び出しを行うように実装します。このメソッドはまた、ビューの `viewActivate` イベントからも呼び出します。このイベントは、ユーザーがビューを前方または後方にナビゲートしたとき呼び出されます。

1. <fx:Script>ブロック内に次のメソッドを追加します。

```
public function refresh(): void {
    getTrends();
}
```

3. ファイル上部にある<s:View>タグに、`viewActivate="refresh()"`を追加します。

サービスから返されたデータをリストにバインディングする

サービスが呼び出されたらつねに、List にはサービスから返された新しいデータを表示させます。

1. 開いた<s:List>タグの<と>の間にカーソルを置きます (<s:List left="" right="" ...この間>)。
2. [データ]→[データにバインド]を選択します。
3. [既存の呼び出し結果]が選択され、ドロップダウンメニューでは既存のコールリスポンダー

(getDataResult か getTrendsResult)がすでに選ばれているはずで

4. (ライブデータの場合のみ)[データプロバイダー]で trends の配列 (trends[])を選択します。
5. [ラベルフィールド]では name を選択します。
6. OK をクリックします。

#### 演習4: アプリケーションのデスクトップでのデバッグ

Trends ビューのコードの記述が終わったので、アプリケーションを実行する準備に移りましょう。最終的には複数の実機でテストするとしても、開発中はコンピュータで手早くテストする方が便利です。実際のデバイスでアプリケーションを実行しデバッグする方法は演習8で見えていきます。

#### 起動の構成を作成しデバッグを開始する

アプリケーションの実行方法を Flash Builder に指定するには、起動の構成を作成する必要があります。

1. [実行]→[デバッグの構成]を選択します。
2. 左のリストにある[モバイルアプリケーション]をダブルクリックします。すると新しい構成が作成できるダイアログが開きます。
3. ダイアログの上部にある[名前]フィールドを TwitterTrends Desktop に変更します。
4. [プロジェクト]フィールドで TwitterTrends が選択されていることを確認します。
5. [起動方法]で[デスクトップ上]を選択し、シミュレートするデバイスに Google Nexus One を選びます。
6. [デバッグ]をクリックします。
7. \_TwitterTrends.mxml の保存が求められたら、[OK]をクリックします。

アプリケーションがデスクトップの AIR Debug Launcher (ADL) で起動し、Twitter トレンドのリストを表示します。

#### ブレークポイントの設定

1. Flash Builder に切り替えます。
2. \_TwitterTrends.mxml の refresh() メソッドのコードを見つけます。
3. refresh() メソッド内の getData() か getTrends() 呼び出しのコードの左のグレーの線上でダブルクリックします。

ブレークポイントを示す青い点が現れます。

#### アプリケーションのテスト

1. ADL で実行中のアプリケーションに戻ります。
2. ADL の [Device] メニューで、[Rotate Right] を選択します。するとウィンドウが横方向に切り替わり、コントロールはそのウィンドウに合うようにサイズが変わります。

3. リストをマウスでドラッグすると、リストを”投げる”ことができます。これはタッチスクロールのシミュレーションです。
4. アクションバーの更新アイコンをクリックします。すると Flash Builder が、Flash デバッグパースペクティブに切り替えてもよいかどうかをたずねるダイアログを表示します。[常にこの設定を使用する]を選択して、[はい]をクリックします。
5. Flash デバッグパースペクティブで新しいパネルが表示されます。

[デバッグ]パネルには現在のコールスタックが表示されます。スタックの最上位は停止した refresh() メソッドです。コールスタックのほかの呼び出しをクリックすると、ほかのメソッドに移動します。

[変数]パネルには現在のスコープでの変数の値と現在のオブジェクト (this) が表示されます。this の左にある三角形をクリックすると、メンバー変数やそのほかの値が見られます。

[ブレークポイント]パネルには現在設定しているブレークポイントが表示されます。

[式]パネルでは、停止した時点で監視したい値の式が作成できます。

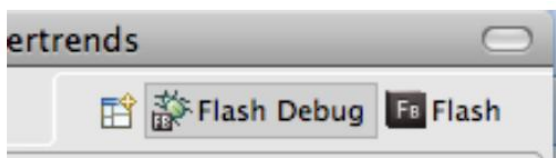
6. [デバッグ]パネルで[ステップイン]ボタンを何回かクリックします。コードの実行が一行ずつ移っていくのが分かります。



7. [再開]ボタンをクリックし、ADL のウィンドウに切り替えます。アプリケーションは実行をつづけます。



8. ADL を終了します。
9. Flash Builder の右上隅にある [Flash] ボタンをクリックすると、Flash 開発パースペクティブに戻ります。このボタンを表示するには、パースペクティブボタンを含むタブのサイズを変える必要があるかもしれません ([Flash デバ...] の左のタブの端を左にドラッグします)。また [ウィンドウ] → [パースペクティブを開く] → [Flash] の選択で切り替えることもできます。



10. refresh()メソッド内のブレークポイント上でダブルクリックしてブレークポイントを削除します。

## 演習5:トレンドのツイートを表示するビューの追加

最初のビューでトレンドのリストが表示できるようになったので、次はユーザーがトレンドをタップしたらそのトレンドのツイートを見られるようにしていきましょう。そのためには新しいビューを作成し、トレンドに関する検索クエリにつなげる必要があります。これは Trends ビューで行った処理と似ています。

### TweetsView の作成

1. [ファイル]→[新規]→[MXML コンポーネント]を選択します。
2. [パッケージ]フィールドを views に設定します。
3. [名前]フィールドを TweetsView に設定します。
4. [レイアウト]はなしのままにしておきます。
5. [ベース]フィールドには spark.components.View が設定されていることを確認します。
6. [終了]をクリックします。
7. ビュー全部を埋める List を追加します。これは演習2で行ったデザインモードでの同じ手順で行えます。また、`</s:View>`終了タグの前に次のコードを入力することでも行えます。

```
<s:List left="0" top="0" right="0" bottom="0">
</s:List>
```

### TweetsView 用のデータを取得する

1. 上部の`<s:View>`タグの title プロパティを"Twitter for {data}"に変更します。この後の演習では、Trends ビューでユーザーが選択したトレンドを、TweetsView の data プロパティとして渡す方法を見ていきます。
2. Trends ビューのときと同様の方法で、ツイートを検索するサービスを次のように作成します。

静的なデータ	ライブデータ
<ol style="list-style-type: none"><li>1. [データ]→[XML に接続]を選びます。</li><li>2. [パス]フィールドの右にある[参照]ボタンをクリックし、src/sampleddata フォルダにある search.xml ファイルを選択します。</li><li>3. [サービス名]を TweetsService に設定します。後はそのまま変更せず[終了]をクリックします。</li><li>4. 下部にある [データとサービス]パネルに</li></ol>	<ol style="list-style-type: none"><li>1. [データとサービス]パネルで TwitterService を右クリックして表示されるコンテキストメニューから[プロパティ]を選択します。</li><li>2. [操作]の表の上にある[追加]ボタンをクリックします。</li><li>3. [名前]に入力されている Operation1 を getTweets に変えます。</li></ol>

TweetsService の `getData()` が表示されるので、右クリックして表示されるコンテキストメニューから[サービスの呼び出しを生成]を選択します。新たに生成された `getData()` 関数が表示されます。

5. コード内の `getData()` 関数の名前を `getTweets()` に変更します。

```
protected function getTweets()
{
    getDataResult.token =
        tweetsService.getData();
}
```

4. 操作の URL を <http://search.twitter.com/search.json> に変えます。
5. [パラメーター]の表の上にある[追加]ボタンを2度クリックし、パラメータを2つ追加します。
6. パラメータの[名前]を `q` と `result_type` に変更します。各名前を入力した後には必ず実行キーを押して入力を確定させます。
7. [終了]をクリックします。
8. [データとサービス]パネルで `getTweets()` を右クリックし、表示されるコンテキストメニューで[戻り値の型を設定]を選びます。
9. [サンプルデータから戻り値の型を自動検出]が選ばれていることを確認し、[次へ]をクリックします。
10. [値を入力]列で、パラメータ `q` の値を `Flex` に、パラメータ `result_type` の値を `recent` に設定します。
11. [次へ]をクリックします。
12. [作成する新規データ型の名前を入力]で、`Tweets` を入力します。
13. `results` プロパティの型の名前を `Results[]` から `Tweet[]` に変えます。
14. [終了]をクリックします。
15. `getTweets()` のコンテキストメニューから[サービスの呼び出しを生成]を選択します。新たに生成された `getTweets()` 関数が表示されます。

ユーザーがビューをナビゲートするか更新したときサービスを呼び出す

これは Trends ビューのときと同じように、`refresh()`メソッドを実装することで実現します。

1. `<fx:Script>`ブロック内に次のメソッドを追加します。

静的なデータ	ライブデータ
<pre>public function refresh(): void {     getTweets(); }</pre>	<pre>public function refresh(): void {     getTweets(String(data), "recent"); }</pre>

```
} }
```

2. <s:View>タグに viewActivate="refresh()"を追加します。

ライブデータを使用するアプリケーションでは、TweetsView の data プロパティを Twitter の検索サービスに渡します。この後の演習ではこの data プロパティを TweetsView に渡す方法を見ていきます。

静的なデータを扱うアプリケーションでは、実際にはバックエンドに照会しないので、パラメータを渡す必要はありません。つねにダミーデータを取得することになります。

サービスが返したデータをリストにバインドする

1. カーソルを開いた<s:List>タグの間に置きます (<s:List left="0" ...>の間をクリックする)。
2. [データ]→[データにバインド]を選択します。
3. [既存の呼び出し結果]がすでに選ばれており、ドロップダウンメニューではコールレスポンス (getDataResult か getTweetsResult) が選ばれているはずです。
4. (ライブデータのみ)[データプロバイダー]で[results[]]配列を選びます。
5. [ラベルフィールド]で text を選びます。
6. [OK]をクリックします。

これで TweetsView がデータを受け取り、それを表示する設定が行えました。しかしユーザーにはまだここまで移動できる方法を提供していません。次の演習ではこのナビゲーションを設定します。

## 演習6:ビューのナビゲーションの設定

TweetsView が作成できたので、次はユーザーが最初のビューのトレンドをタップしたとき、アプリケーションが TweetsView にナビゲートするように設定する必要があります。

ユーザーが、\_TwitterTrends.mxml の List コンポーネントをタップすると、List は選択が変更されたことを示す change イベントを送出します。このイベントを処理すると、ViewNavigator に対しビュースタック(ビューの層)に TweetsView を新たにプッシュする(層の一番上に押し上げる)よう伝えることができます。

ViewNavigator はスタック内の View のセットを管理するコンポーネントです。アプリケーションのスタート時、スタックにはアプリケーションの firstView しかありません。新しいビューにナビゲートしたいときには、ViewNavigator の pushView() を呼び出すことで、新しいビューをスタックにプッシュします。ユーザーがデバイスの戻るボタンをタップすると、現在のビューが自動的にスタックからポップオフされます(なくなります)。最上位にあるビューのスタックからのポップは popView() で行うこともできます。

通常 ViewNavigator は自分で作成する必要はありません。これは MobileApplication の作成時に1つ提供されます。

ビューのセットはスタックとして維持管理されますが、メモリ内に実際に存在するのは1つのビュー(現在アクティブなビュー)だけで、前のビューは自動的に破棄されます。しかし破棄されたビューの data プロパティが保持されるので、ユーザーが前のビューに戻っても、ViewNavigator がその適切なデータを使ってビューを再インスタンス化します。

\_TwitterTrends で change イベントを追加して TweetsView にナビゲートする

Flash Builder のコンテンツアシスト機能を使うと、change イベントハンドラは簡単に作成できます。

1. エディタを \_TwitterTrends.mxml に切り替えます。
2. <s:List>タグに移動し、開始タグの>の直前をクリックします(つまり labelField="name"の後)。
3. 半角スペースにつづけて ch まで入力すると、プロパティとイベント名を示すコンテンツアシストポップアップが表示され、change がハイライト表示されます。実行キーを押して change を選びます。
4. [Change ハンドラーを生成]というポップアップが表示されるので、実行キーを押します。すると Flash Builder が List に "list" という id を与え、list\_changeHandler メソッドを作成します。
5. 作成された list\_changeHandler メソッドに次のコードを記述します。

```
navigator.pushView(TweetsView, list.selectedItem.name);
```

これは ViewNavigator に対し、TweetsView のインスタンスを作成し、選択されたトレンドの名前を TweetsView インスタンスの data プロパティとして、TweetsView インスタンスに渡すように伝えます。前の演習では data プロパティを TweetsView の title にバインディングし、(ライブ API を使っている場合には) data プロパティを使ってトレンドに関するツイートを Twitter の検索サービスに照会しています。

今の時点でアプリケーションを実行し、最初のビューでトレンドをタップすると、リストにツイートが表示されます。しかしリストには各ツイートの行が1行表示されるだけで、それを投稿した人の画像も名前も表示されません。次の演習ではこの設定を行っていきます。

## 演習7:リストのアイテムレンダラの設定

前の演習の最後で述べたように、リストにはデータを表示する方法を教える必要があります。Flex ではこれを、リストのアイテムレンダラを作成することで行います。

デスクトップの Flex アプリケーションでは通常、ItemRenderer をベースにしたカスタムの MXML コンポーネントを作成してこれを実現しますが、モバイルアプリケーションで MXML を使ってアイテムレンダラを作成すると、リストをスクロールするときにパフォーマンスの問題が発生する可能性があります。したがってわれわれは、モバイルの Flex アプリケーションでは、ActionScript ベースのアイテムレンダラの使用を推奨します。

“Hero”では ActionScript ベースの MobileIconItemRenderer クラスが提供されています。このアイテムレンダラを使用すると、ヘッダフィールドを表示し、その下にオプションのメッセージフィールドが表示できます。またオプションで、各アイテム内のテキストの左にはアイコンや画像も表示できます。

MobileIconItemRenderer がみなさんの要求を満たさない場合には、汎用的な基本クラスの MobileItemRenderer のサブクラスを作成し、みなさん独自のコントロールを ActionScript で追加しレイアウトします。その方法を示すのはこのチュートリアルを超えますが、今後そのサンプルを示す機会があるかもしれません。

### モバイルアイテムレンダラの作成

Flash Builder “Burrito”の新しいコードテンプレート機能を使用すると、リストに手早く MobileIconItemRenderer が追加できます。

1. TweetView.mxml に切り替えます (似たような List のコードのある \_TweetView.mxml ではありません)。
2. List から labelField="text"プロパティを消去します。このテキストはアイテムレンダラで以下のように置き換えます。
3. カーソルを List の<s:AsyncListView/>タグの終わりの直後に置き、実行キーを押します。
4. Mobile と入力します (前に<はつけません)。
5. Ctrl-スペースを押します。

#### 訳者注:

訳者の Mac では Ctrl-スペースの同時押しで Spotlight が表示されたので、[システム環境設定]の[キーボード]→[Spotlight]で、Spotlight に関連づけられている ^スペースの選択を解除しました。

Flash Builder には MobileIconItemRenderer 用コードテンプレートが備わっているので (Mobile で始まるテンプレートはこれだけです)、コードの基本構造をみなさんに代わって記述してくれます。

6. アイテムレンダラのコードを次のように記述します。

```
<s:itemRenderer>
    <fx:Component>
        <s:MobileIconItemRenderer
            iconField="profile_image_url"
            iconWidth="80"
            iconHeight="80"
            labelField=""
            messageField="text"
            height="110"
            verticalAlign="top"/>
    </fx:Component>
</s:itemRenderer>
```

Field の名前をついたプロパティでは、メッセージとアイコンとして使用する各データアイテムのメンバーを指定します。labelField の代わりに messageField を使用すると、テキストを複数行でラップできます。labelField (これはアイテムレンダラの headerField をマッピングします) ではデフォルトで、テキストは太字で表示され、ラップされません。

ただし“Hero”のこのプレビューリリースでは、ツイートが高くなっても (行数が増えても)、アイテムレンダラは自動的に垂直方向に大きくなりません。したがってここではその高さを修正しています。最終リリースでは、レンダラがその内容を収めるために自動的に大きくなるので、この高さの修正は必要なくなります。

7. Flash Builder のツールバーにある [実行] ボタンをクリックし、アプリケーションをデスクトップで実行します。
8. 最初のビューでトレンドをクリックします (静的なデータを使用している場合には “Inception” をクリックします)。そのトレンドに関するツイートが表示されます。
9. Trends ビューに戻るには、ADL のメニューから [Device] → [Back] を選択します (または Ctrl/Cmd + B を押します。ADL のメニューが表示されない場合には、一度別のアプリケーションに切り替え、再度 ADL に切り替えます)。
10. ADL を終了します。

中には完全に表示されないツイートがあります。プレビューリリースの `MobileIconItemRenderer` はその内容の高さに合わせてサイズを変更しません。これが高さを明示的に追加した理由です。しかしこれは最終リリースでは修正されます。

それぞれのツイートにユーザー名を追加する

1. 作成した `MobileIconItemRenderer` タグ内で、`messageField="text"` を `messageFunction="getTweetText"`に変更します。
2. `MobileIconItemRenderer` タグ内に開始と終了タグを加え、スクリプトブロックを次のように追加します。

```
<s:MobileIconItemRenderer
  iconField="profile_image_url"
  iconWidth="80" iconHeight="80"
  labelField="" messageFunction="getTweetText"
  height="110" verticalAlign="top">
  <fx:Script>
    <![CDATA[
      private function getTweetText(item: Object): String
      {
          return item.from_user + ": " + item.text;
      }
    ]]>
  </fx:Script>
</s:MobileIconItemRenderer>
```

訳者注:

具体的に言うと、`MobileIconItemRenderer` タグの最後の `"top"/>` の `/` を削除し、`</s:MobileIconItemRenderer>`を追加します。そのタグ内に、上記コードを追加します。

アプリケーションを再度実行すると、今度は各ツイートの前にユーザー名が表示されます。

## 演習8: デバイス上でのアプリケーションのデバッグ

アプリケーションを完成させる前に、実機でこれを実行してみましょう。そのためには Android 2.2 (“Froyo”)が稼働するデバイスが必要で、コンピュータに USB ケーブルで接続します。デバイスをお持ちでない方は次の演習に進んでください。


### デバイスの設定

1. デバイスで USB デバッグを有効にします。
  - (ア) Home ボタンをタップし Home 画面に移動します。
  - (イ) Menu ボタンをタップし[設定]を選びます。
  - (ウ) [アプリケーション]をタップし、[開発]をタップします。
  - (エ) [USB デバッグ]が選択されていることを確認します。
2. デバイスを USB ケーブルを使ってコンピュータに接続します。
3. 画面最上部にある通知領域をプルダウンします。すると“USB デバッグが接続されました”と表示されているはずで。
  - (ア) “USB デバッグが接続されました”をタップします。
  - (イ) “スリープモードにしない”を含むオプションが表示された場合には、これをタップし有効にします。
4. デバイスに以前の AIR 2.5 for Android をインストールしている場合には、まずこれをアンインストールします。
  - (ア) Home をタップし、Menu、つづいて[設定]をタップします。
  - (イ) [アプリケーション]をタップし、[アプリケーションの管理]をタップします。
  - (ウ) Adobe AIR をタップします。
  - (エ) [アンインストール]をタップします。
5. ライブの Twitter API を使用している場合は、デバイスが WiFi かモバイルネットワーク上に存在することを確認してください。

### デバイス上での実行

1. Flash Builder のツールバーにある[実行]ボタンの右の矢印をクリックし、[実行の構成]を選びます。
2. 左のリストで前に設定した TwitterTrends Desktop を選択し、上にある[現在選択されている起動構成の複製]ボタン([X]の左)をクリックします(TwitterTrends Desktop を右クリックし[複製]を選ぶ方が早い)。
3. [名前]を TwitterTrends Device に変更します。
4. [起動方法]を[デバイス上]に変更します。
5. [適用]をクリックし[実行]をクリックします。

初めて起動するときには、Flash Builder がデバイスに AIR ランタイムをインストールしてから、アプリケーションをインストールします。起動の進捗状況は、Flash Builder の右下部のプログレスバーで表示されます。

Launching TwitterTrends Device: (57%) 

本訳者注:

Flex Builder がインストールする AIR ランタイムはバージョン 2.5.0.1660 です。翻訳時点での最新版は 2.5.1.1774 です。

起動したアプリケーションでは、デスクトップのときと同じようにトレンドのリストをスクロールし、トレンドをタップしてそのツイートを見、デバイスの戻るボタンで移動できます。またデバイスの向きを横方向に変えると、アプリケーションもそれに合わせて表示を変えます。(ライブデータを使用しているときには) [更新] ボタンをタップすると、最新のツイートが得られます。

#### デバイスのデバッグ用設定

デバイスでのデバッグでは、AIR ランタイムはネットワークを通して Flash Builder 内のデバッガとやりとりします。これはつまり、デバイスはコンピュータの IP アドレスでコンピュータにアクセスできるネットワークに WiFi 接続している必要があるということです。

1. デバイスを WiFi 接続させます。
  - (ア) Home ボタンをタップします。
  - (イ) Menu ボタンをタップし、[設定]をタップします。
  - (ウ) [無線とネットワーク]をタップします。
  - (エ) 選択されていない場合には[Wi-Fi]を選択します。
  - (オ) [Wi-Fi 設定]をタップします。
  - (カ) まだ WiFi に接続していない場合には、接続するネットワークをタップします (MAX では、OfficialMAX2010Wifi ネットワークを使用します)。
2. ラップトップも同じネットワークに接続する必要があります (MAX では同じ WiFi ネットワーク上にあることが必要です。ほかの場合では、デバイスが接続する WiFi ネットワークからアクセスできるネットワーク上にあることが必要です)。
3. Windows Vista や Windows 7 を使ってデバッグするには、プレビューリリースの問題からファイアウォールを一時的に無効にする必要があります。Windows XP ではポート 7 と 7935 を開きます。

## ブレークポイントの使用

1. Flash Builder で、\_TwitterTrends.xml の refresh()メソッドにブレークポイントを設定します（演習4で見た方法です）。
2. [デバッグ]ボタンの右の三角マークをクリックし、TwitterTrends Device を選びます。デバイスで実行されていたアプリケーションが終了し、デバッグモードで再スタートします。
3. アプリケーションの再スタート時にデバイスで IP アドレスのダイアログが表示される場合には、デバグがコンピュータに接続できていません。前の“デバイスのデバッグ用設定”を見直してください。
4. アプリケーションがスタートすると、ブレークポイントにヒットします（refresh()は最初のビューがアクティブ化される時呼び出されます。viewActivate=”refresh()”）。

Flash Builder のデバイスでのデバッグ機能には、演習4のデスクトップと同じように、すべての機能にアクセスできます。ただし関数内のステップ操作や[変数]でのオブジェクトの中身のドリルダウンなど、中には遅くなる可能性のある操作もあります。

5. Refresh()メソッドからブレークポイントを、グレーの線上でダブルクリックすることで削除します。

## 追加演習

(最終的なアプリケーションをパッケージ化する方法は演習 11 で取り上げています)

## 演習9: 選択したユーザーの情報を表示するビューの追加

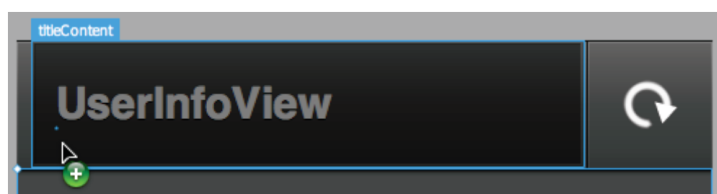
アプリケーションはいよいよ、任意のツイートを投稿したユーザーの情報を表示するビューを追加して完成です。

このビューはこれまでのビューよりレイアウトが少し複雑で、アクションバーをカスタマイズして選択したユーザーの画像と名前を表示し、ビューのコンテンツにユーザーの位置情報と Web サイトを表示するラベルを加えるほか、ユーザーの最近のツイートのリストも表示します。

### ビューを作成しアクションバーをカスタマイズする

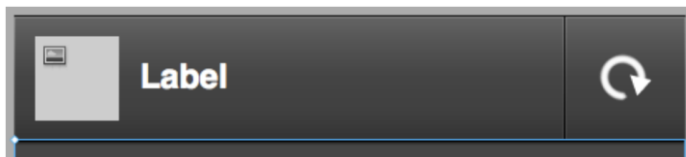
これまでビューのアクションバーは変更しなかったため、MobileApplication のコンテンツがそのまま表示されていました (titleContent プロパティの場合には、現在のビューのタイトルが表示されます) が、今度はデフォルトの titleContent をツイートを投稿したユーザーの画像と名前で置き換えます。

1. TweetView のときと同様に[ファイル]→[新規]→[MXML コンポーネント]を使って、views パッケージ内に UserInfoView という名前の新しい View を作成します。
2. デザインモードに切り替えます。
3. [コンポーネント]パネルの[レイアウト]フォルダから Spacer をアクションバーの titleContent 領域にドラッグします。これによってデフォルトの titleContent が置き換えられ、表示されなくなります。



4. プロパティパネルで[幅]を 15 に設定します。
5. [画像]コンポーネントを Spacer の右の titleContent にドラッグし、[幅]と[高さ]を 60 に設定します (Note: プレビューリリースの問題により、titleContent コンテナボックスは正しく表示されないかもしれません)。
6. Spacer をもう1つ、[画像]の右にドラッグし、[幅]を 15 に設定します。
7. Spacer の右に Label をドラッグします。
8. [プロパティ]パネルの[太字]ボタンをクリックします。

ここまでの作業でアクションバーは次の図のようになっているはずです。

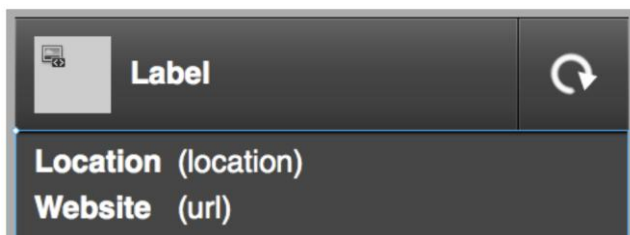


本訳者注:

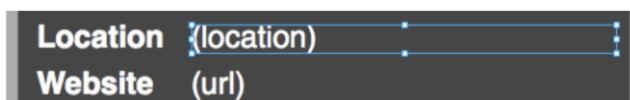
訳者の作業では、上図の”Label”は“ラベル”とカタカナで表示されています。また[画像]コンポーネントは、[コンポーネント]パネルで[画像]という日本語が使われているので、そのまま[画像]コンポーネントと記していますが、原書では Image です。

ビューコンテンツのレイアウト

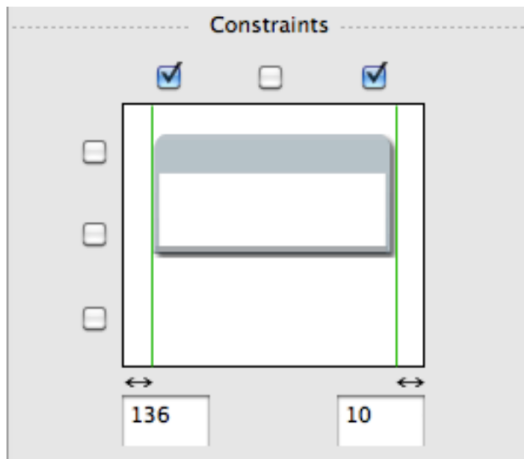
1. [コンポーネント]パネルから[Label]を4つドラッグし、下図のように並べます。



2. 各ラベルをダブルクリックして、テキストを上図のように設定します。右のラベル((location)と(url))は後からデータをバインドするプレースホルダーです。
3. 左の2つのラベルを選択し、[プロパティ]パネルの[太字]ボタンをクリックします。
4. 右の“(location)”ラベルを選択します。  
(ア) ラベルの右端をビューの右端近くまで、余白を残して伸ばします。



- (イ) [プロパティ]パネルの[制約]で、下図のように左と右端を固定します(左の数値はみなさんの数値と同じでないかもしれませんが)



(ウ) [プロパティ]パネルの表示を[アルファベット順ビュー]表示([プロパティパネル]右上隅のボタン)に切り替え、[maxDisplayedLines]を 1 に設定します。

(エ) [プロパティ]パネルの表示を[標準ビュー]表示に戻します。

5. 手順4を右の”(url)”ラベルについて繰り返します。

6. ラベルの下に List をドラッグします。ビューの左右と下端付近まで一杯に伸ばし、四隅を[制約]で固定します。

#### ビューで使用するデータの取得

このビューには2つのデータセットが必要で、異なるサービスから取得します。1つめはユーザーのプロファイル情報です。これはアクションバーの画像とその下の位置情報(location)と Web サイト(url)に使用します。2つめのデータセットはユーザーの最近のツイートで、これはその下の List で使用します。

静的なデータを使用している場合には、ユーザー情報と最近のツイート情報を入れた2つの静的なデータファイルを使用します。ライブデータを使用している場合には、ユーザー情報用の新しいサービス呼び出しにアクセスしますが、最近のツイート情報の取得には前に作成した `getTweets()` サービス呼び出しを使用します。

静的なデータ	ライブデータ
<ol style="list-style-type: none"> <li>1. [データ]→[XML に接続]を選択します。</li> <li>2. [パス]フィールドの右の[参照]をクリックし、src/sampledata フォルダ内の userinfo.xml ファイルを指定します。</li> <li>3. [サービス名]を UserInfoService にします。そのほかはデフォルトのままにして[終了]をクリックします。</li> </ol>	<ol style="list-style-type: none"> <li>1. [データとサービス]パネルで TwitterService を右クリックし[プロパティ]を選択します。</li> <li>2. [操作]の右にある[追加]ボタンをクリックします。</li> <li>3. [名前]の Operation1 を getUserInfo に変更します。</li> <li>4. [ 操 作 ] の URL に</li> </ol>

<p>4. 下の[データとサービス]パネルに現れる UserInfoService の getData()を右クリックして、[サービスの呼び出しを生成]を選択します。新しい getData()関数が生成されます。</p> <p>5. 関数名を getData から getUserInfo に変えます。</p> <p>6. getResult の上で右クリックし、[リファクタリング]→[名前を変更]を選択します。</p> <p>7. 名前を getUserInfoResult に変更して [OK]をクリックします。この処理によって getResult を使用していた2か所で getResult が getUserInfoResult に変更されます。</p> <p>8. ここまでの手順 1 から 7 を searchUserTweets.xml についても繰り返します。サービス名は UserTweetsService にし、新しい getData()関数は getTweets() に名前を変え、[リファクタリング]→[名前を変更]を使って getResult を getResult に変えます。</p>	<p><a href="http://api.twitter.com/1/users/show.xml">http://api.twitter.com/1/users/show.xml</a>を入力します。</p> <p>5. [パラメーター]表の上の[追加]ボタンをクリックします。</p> <p>6. [名前]を screen_name に変更します。パラメータ名の入力後は必ず実行キーを押して確定させます。</p> <p>7. [終了]をクリックします。</p> <p>8. [データとサービス]パネルで getUserInfo()を右クリックし[戻り値の型を設定]を選択します。</p> <p>9. [サンプルデータから戻り値の型を自動検出]が選ばれていることを確認し、[次へ]をクリックします。</p> <p>10. [値を入力]列でパラメーター screen_name を Adobe に設定します。</p> <p>11. [次へ]をクリックします。</p> <p>12. [作成する新規データ型の名前を入力]にはデフォルトで User が入力されているはずです。[終了]をクリックします。</p> <p>13. getUserInfo()メソッドのコンテキストメニューから[サービスの呼び出しを生成]を選択します。新たに getUserInfo()関数が生成されます。</p> <p>14. 既存の getTweets()メソッドに関するサービス呼び出しも生成させます (getTweets()メソッドのコンテキストメニューから[サービスの呼び出しを生成]を選択します)。このサービスは任意のユーザーのツイートの検索に使用します。</p>
--	--

ユーザーがビューをナビゲートするか更新したときサービスを呼び出す

Trends ビューのときのように、refresh()メソッドを実装します。

1. <fx:Script>ブロック内に次のメソッドを追加します。

静的なデータ	ライブデータ
<pre>public function refresh(): void {     getUserInfo();     getTweets(); }</pre>	<pre>public function refresh(): void {     getUserInfo(String(data));     getTweets("from:" + String(data), "recent"); }</pre>

2. ファイル冒頭の<s:View>タグに viewActivate="refresh()"を追加します。

サービスが返したデータを UI にバインドする

1. カーソルを開いた<s:List>タグの<と>の間に置きます。
2. [データ]→[データにバインド]を選びます。
3. [既存の呼び出し結果]で getTweetsResult を選びます。
4. (ライブデータのみ) [データプロバイダー]で results[]配列を選びます。
5. [ラベルフィールド]で text を選択します。
6. [OK]をクリックします。
7. 演習7の例にしたがい、アイテムレンダラの messageField としてテキストを取得し、labelField を空に設定します(ツイートはすべて同一ユーザーからのものなので、アイコンの指定やユーザー名の追加は不要です)。アイテムレンダラのコードは次のようになります。

```
<s:MobileIconItemRenderer labelField="" messageField="text"
height="110" verticalAlign="top"/>
```

この後の手順では、手作業でコードを入力し、残りの UI 要素をデータにバインドします。

1. <s:titleContent> タグ内の Image に移動し、source を {getUserInfoResult.lastResult.profile\_image\_url}に設定します。
2. <titleContent>内の Label の text を{data}に設定します。このビューには後から選択したユーザーのスクリーン名を data として渡します。
3. プレースホルダーの (location) ラベルで、text プロパティを {getUserInfoResult.lastResult.location}に変更します。
4. プレースホルダーの (url)ラベルで、text プロパティを {getUserInfoResult.lastResult.url}に変更します。

UserInfoView にナビゲートする change ハンドラを TweetsView に追加する

1. エディタを TweetsView.mxml に切り替えます。
2. 前に行ったように(P18 参照)、コンテンツアシストを使って List タグで change ハンドラを生成します。  
このとき List の id は list になっている必要があります。
3. 生成された list\_changeHandler メソッドに次のコードを記述します。

```
navigator.pushView(UserInfoView, list.selectedItem.from_user);
```

ここまでの作業でアプリケーションが実行できます。トレンドをタップしツイートをタップすると、そのツイートを投稿したユーザーの情報に移動できます(静的なデータの場合には、Tweets ビューの最初のツイートをタップすると、そのユーザーのデータに移動できます)。

## 演習10:セッション間でデータを存続させる設定

デスクトップアプリケーションを実行しているときユーザーは、何らかの作業が終わったら明示的にアプリケーションを終了します。これに対し多くのモバイルオペレーティングシステムでは、ユーザーは通常、アプリケーションを明示的に終了させません。たとえば Android では、ユーザーがアプリケーションの最初の画面を見ているとき Home ボタンか戻るボタンを押しても、アプリケーションは終了しません。アプリケーションは単に背後に送られるだけです。その逆に、多くのアプリケーションを実行しているオペレーティングシステムは、背後にあるアプリケーションをユーザーに何の通知もせず終了させます。これはつまり、モバイルアプリケーションでは、ユーザーが次にアプリケーションに戻ってきたとき、それまでの作業が失われないように、状態の保存と回復に注意を払う必要があるということです。

Flex ではこれを自動的に行うセッションキャッシング機能を MobileApplication クラスに用意しています。セッションキャッシングを有効にすると、アプリケーションは終了時、自動的にビュースタックとビューのデータをデバイスの永続的なストレージに保存し、再スタート時に状態を回復します。

### セッションキャッシングを有効にする

1. TwitterTrends.mxml に切り替えます。
2. `<s:MobileApplication>`タグで、プロパティ `sessionCachingEnabled="true"`を追加します。

たったこれだけでセッションキャッシングが有効になります。このアプリケーションでは各ビューの data プロパティが単純なストリングなので、セッションキャッシングも単純ですが、もっと複雑なアプリケーションでは、持続的なマネージャーにデータ型クラスを登録したり、特定のデータ型に関するシリアル化を独自に実装する必要があるかもしれません。

### セッションキャッシングをデスクトップでテストする

1. [実行]ボタンの右の矢印をクリックします。
2. TwitterTrends Desktop を選択します。
3. トレンドをタップし、TweetsView に移動します。
4. ツイートをタップし、UserInfoView に移動します。
5. ADL のウィンドウを閉じます。
6. 再度アプリケーションを起動します。アプリケーションは UserInfoView からスタートし、アプリケーションを終了したときに見ていたユーザーを表示します。
7. ADL で [Device] → [Back] (または Ctrl-B/Cmd-B) を選択します。アプリケーションは、前のセッションでみなさんが見ていたトレンドに関するツイートのリストに戻ります。

セッションキャッシングはビュースタックの全体的な履歴と適切なデータを維持管理します。ユーザーの立場から見ると、アプリケーションは中断したようには見えません。

この機能はデバイス上でもテストできますが、テストするにはアプリケーションを手動で強制終了させる必要があります。Android では、[設定]→[アプリケーション]→[アプリケーションの管理]画面からアプリケーションを選び、[強制終了]をタップします。

実行時にアプリケーションのデータを消去する

セッションキャッシングを有効にしたアプリケーションをテストするときには、アプリケーションを最初のクリーンな状態からスタートさせるため、その実行時にアプリケーションのデータを一掃したいでしょう。

1. [実行]→[実行構成]を選択します。
2. 左のリストで TwitterTrends Desktop を選びます。
3. 右の画面下部の[起動ごとにアプリケーションデータを消去]を選択します。
4. [実行]をクリックします。

するとアプリケーションは最初の Trends ビューから再スタートします。セッション状態の保存に再度戻りたい場合には、[実行構成]ダイアログに戻り、[起動ごとにアプリケーションデータを消去]の選択を解除します。このチェックボックスはデバイスベースの実行構成でも機能します。

## 演習11: リリース用アプリケーションパッケージの書き出し

アプリケーションが完成したら、それをほかの人に提供したり、アプリケーションストアで販売したくなるでしょう。そのためには、モバイルプロジェクトのパッケージ化もできるようになった Flash Builder の[リリースビルドのエクスポート]を使用します。

1. [プロジェクト]→[リリースビルドのエクスポート]を選択します。
2. [プロジェクト]ドロップダウンメニューで TwitterTrends が選ばれていることを確認します。
3. お持ちの場合にはデバイスが接続されていると便利です。
4. [次へ]をクリックします。Flash Builder がプロジェクトの初期ビルドを実行します。
5. 次の画面の[証明書]フィールドの右にある[作成]ボタンをクリックすると、非認証の自己署名の証明書が作成できます。
6. [発行者名]を入力します。
7. [パスワード]と[パスワードの確認]フィールドにパスワードを入力します。
8. [参照]ボタンをクリックして証明書を作成する場所とファイル名を指定します。
9. [OK]をクリックします。
10. [リリースビルドの書き出し]ダイアログの [終了]をクリックします。デバイスを接続していない場合には、ビルドがデバイスにインストールされなかったことが通知されます。

リリースビルドはプロジェクトのルートに TwitterTrends.apk として保存され、デバイスを接続している場合には、デバイスにインストールされます。アプリケーションのデバッグバージョン(以前デバイスでアプリケーションをデバッグしたときや実行したときにインストールされています)の名前は TwitterTrends-debug で、今度のリリースバージョンの名前は TwitterTrends(-debug がついていない)です。

アプリケーションを公に配布するには(Web サイトや Android Market などのアプリケーションストアで)、証明書機関から認証された証明書を取得する必要があり、それを使ってリリースビルドを作成します。

さらに追加演習:

おめでとうございます！ みなさんは初めてのモバイル Flex アプリケーションが作成できました。Flex に馴染まれている方はこのチュートリアルなど朝飯前で、次なる挑戦に指を鳴らしておられることでしょう。そのようなみなさんは以下にトライしてみてください(ライブ API データを使った方がよいでしょう)。

- スプラッシュ画面とアプリケーションアイコンの追加(ヒント: スプラッシュ画面は MobileApplication で、アイコンは AIR XML 記述ファイルで設定します)
- UserInfoView の URL にユーザーの Web サイトをリンクさせる(ヒント: Link というようなコントロールはありませんが、リンクのように見せることは可能です。クリックイベントを処理し、Flash の navigateToURL()メソッドを使用します)
- 任意のキーワードを使ってツイートを検索する方法を追加する(ヒント: アクションバーの titleContent には TextInput コントロールが追加できます)
- Twitter にログインする方法を追加し Twitter ストリームを取得する
- ログインしたらツイートが投稿できるビューを追加する

このほかのチュートリアルやサンプルについては、Adobe Labs の「[Sample Applications and Tutorials](#)」を参照してください。