Javaルーキーでも大丈夫!

勝手に作ろう! iアプリ

第2回 カウンターを使って画面を動かそう

木寺祥友＋株式会社エル・カミノ・リアル
Jump www.ecreal.co.jp

カウンターで画面を動かそう!

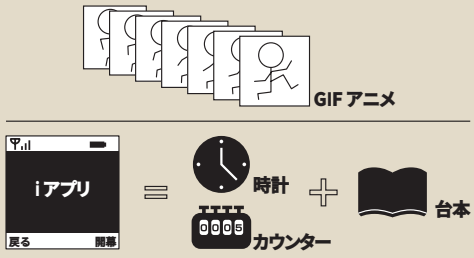
前回はメインクラスの構成を中心にiアプリ作成の基礎を説明したが、クラスやインスタンスなどの概念については理解してもらえただろうか。今回はより具体的なテクニックに入って、カウンターを使ってテキストや画像をコントロールする方法を紹介しよう。

カウンターを使った表現というのは、ウェブサイト上で普及しているFlashと同じようなものと考えてもらってよい。というのも、特定の時間ごとにイベントが発生するようにプログラミングすることで、動きのある表現をする点が似ているからだ。だが残念ながらFlashはケータイ上では動作しないし、かといってGIFアニメーションではファイルのサイズが大きすぎるうえにインタラクティブな動きを出せない。そこでカウンターの出番というわけだ。

それではさっそく、簡単に今回のテクニックの概要を説明しておこう。このテクニックのポイントはShortTimerというタイマーにある。このタイマーは一定時間ごとにイベントを発生させるのだが、このイベントを「カウンターの値を増加させていく」としておき、そしてその値に応じてさらなるイベントを起こすことで、時間制御に近い動きをさせられるわけだ。たとえば0.1秒に1回タイマーイベントを起こしてカウンターが回るようにしておき、そのうえで「カウンターが100に達したら文字を出す」としておけば、タイマーを動かしてから10秒後に文字を表示できる。このほかにもタイマーイベントとカウンターを軸にすればいろいろな変化を時間ごとに与えられるのだ。

それでは、次のページからはMyIntroのソースを見ながら、このテクニックについてよりくわしく解説していこう。

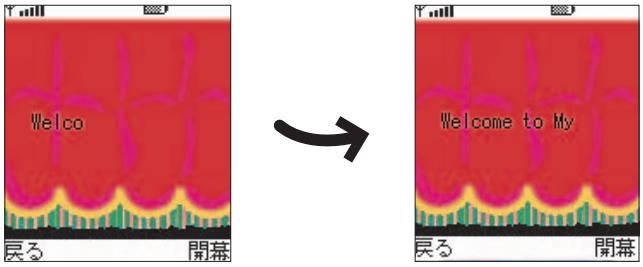
GIFアニメとiアプリ



GIFアニメはパラパラアニメのように、すべての絵が1つのファイル内に入っていて、これをめくることで動きを表現する。そのため、何枚もの静止画が必要となりファイルサイズが大きくなってしまう。これに対してiアプリだと、時計とカウンターと台本、そして素材を用意しておくだけでよい。あとは端末がそれをもとに計算して動きのある画面を作ってくれるのだ。

Flash（フラッシュ）：マクロメディア（株）によるアニメーション作成ツール。プレイヤーは無償で配布されている。

カウンターで文字を操作する



カウンターで文字を操作する部分は、大まかに3つの構造に分けられる。タイマーの起動と、カウンターの増加、そしてテキストの表示だ。タイマーイベントによってカウンターが増加し、それに応じてテキストが表示されるのだ。

■タイマーを起動する【171-175行目】

160行目から176行目までが、ソフトキーが押されると動き出す部分だ。そのなかでカウンターとして使うtmCntを0に初期化し、174行目の“st=ShortTimer.getShortTimer（This, 0, ITV, true）”でタイマーを準備している。ここの3番目の引数ではタイマーイベントを起こす間隔を1/1000秒単位で指定するのだが、今回はすでにITVに100が設定されているため、0.1秒ごとにイベントが発生するようになっている。175行目の“st.start()”で実際にタイマーを起動している。

■タイマーの2重起動の禁止【164-169行目】

開幕ボタンを続けて2回押すと、タイマーが起動している上に同じタイマーを作ろうとしてエラーが発生する。これを防ぐための処置だ。

■タイマーイベントの処理【143-152行目】

```
if （type ==display.TIMER_EXPIRED_EVENT）
```

イベントを処理するprocessメソッドのなかで、上記の条件を満たすものがタイマーイベントになる。つまり、タイマーイベントの処理はここに記述する。146行目の“tmCnt++”は、自分自身であるtmCntに1を加えるという意味だ。つまり、タイマーイベントが発生するごとにカウンターを進めているわけだ。最後に、文字などの表示を行う独自メソッド“opening()”を呼んでいる。

■文字を操作する部分【185-196行目】

ここが、149行目の“opening()”により呼び出されるopeningメソッドの中身の部分だ。ここでは文字表示処理を行っている。以下、それぞれを見ていこう。

■文字を表示する必要性のチェック【189行目】

ここで出てきているmsgCntには、次の文字を表示するカウンターの間隔がセットしてある。つまり、毎回文字を描画するのではなく、msgCntごとに次の文字を表示すればよい。そこで、割り算の余りを求める演算子「%」を使って、“（tmCnt % msgCnt）”の結果が0でなければ処理を中止している。結果的に、msgCntごとに文字の表示処理が行われることになる。

■カウンターに合わせた文字の表示【193行目】

drawStringが文字の表示を、substringが文字列TopFrm.omの中身を抜き出しているメソッドだ。たとえば“drawString（TopFrm.om.substring（0, 10））”で、TopFrm.omに入っている文字列の1文字目から10文字目までが抜き出されて表示される。そこで、193行目にあるようにsubstringの2番目の引数をカウンターと連動させて1つずつ増やすことでTopFrm.omに入っている文字を1つずつ増やしながら表示できるのだ。

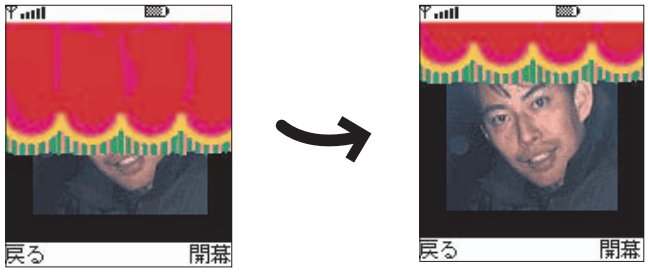
OpenFrm.java 142-196行目

```
142   // タイマーイベント
143     if ( type == Display.TIMER_EXPIRED_EVENT )
144     {
145       // カウントを増やす
146       tmCnt++;
147
148       // 開幕処理
149       opening();
150
151       return;
152     }

160      // ソフトキー2＆キープレスイベント
161     else if ( param == Display.KEY_SOFT2 && type
        == Display.KEY_PRESSED_EVENT )
162     {
163       // 開幕
164       // 既に開幕の場合は初期化
165       try
166       {
167         st.stop();
168         st.dispose();
169       }
170       catch ( Exception e ) {}
171       // 開幕
172       tmCnt = 0;
173       repaint();
174       st = ShortTimer.getShortTimer( this, 0, ITV,
          true );
175       st.start();
176     }

185     // メッセージの表示
186     if ( tmCnt < msgTm )
187     {
188       // 表示対象カウントのチェック
189       if ( ( tmCnt % msgCnt ) != 0 ) { return; }
190
191       // 表示
192       gph.lock();
193         gph.drawString( TopFrm.om.substring( 0,
          tmCnt / msgCnt ), msgX, msgY );
194       gph.unlock( false );
195       return;
196     }
```

■：変数名　■：コメント　■：外部ファイル名やカスタマイズ可能なところ

イベント：ボタンを押したときや、一定の時間が経過したときのように、「何々した場合」を表す。Javaはイベントに合わせて動き出すように記述できる。

カウンターで画像を操作する



それでは文字を動かしていたテクニックを用いて、今度は画像を動かしてみよう。MyIntroのオープニングでは、どん帳が上がるような効果を出しているが、iアプリにそのような機能があってスムーズに動かしているわけではない。その実はカウンターによって毎回位置を変えて画像を貼りなおすことで、あたかも動いているかのように見せているのである。

■画面全体をクリアする【117行目】

clearRectメソッドは画面を背景色で塗りつぶすメソッドだ。始めの2つの引数が塗りつぶす範囲の左上の座標を、次の2つが範囲の大きさを表している。ちなみに、iアプリでは画面座標の基点は左上で、縦座標の値が大きくなるほど下の方向を指し、横座標の値が大きくなるほど右の方向を指す。

■幕画像の表示を開始するカウンターのチェック【128行目】

タイマーイベントが重なり、カウンターtmCntがdonTmの値を超えるとif文のなかに入り、どん帳画像の座標を計算をし直すことになる。それまでは132行目で描画し直しても3番目で指定する縦座標wkYが変わらないので動かない。

■カウンターに合わせた幕画像の表示【127-132行目】

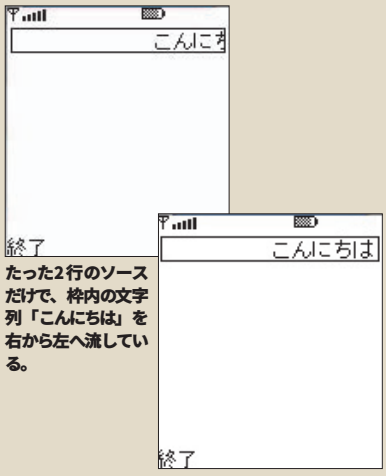
ここでは画像位置の高さであるwkYを計算している。カウンターが回るとtmCntは大きくなるがdonTmは変わらないために、0.1秒ごとにwkYのマイナス値が大きくなっていく。それにより132行目でどん帳を書き直していくときに画像の縦座標の値が少しずつ変更されて、どん帳が上がっていくように見えるのだ。

OpenFrm.java 116-131行目

```
        // 画面クリア
117     g.clearRect( 0, 0, dW, dH );

126     // 緞帳の表示
127     int wkY = 0;
128     if ( tmCnt > donTm )
129     {
130       wkY = ( donTm - tmCnt ) * donV;
131     }
132     g.drawImage( imgD, 0, wkY );
```

■：変数名　■：コメント　■：外部ファイル名やカスタマイズ可能なところ

Panelならもっと簡単にできるけど…



たった2行のソースだけで、枠内の文字列「こんにちは」を右から左へ流している。

```
// ティッカー
Ticker tck = new Ticker( "こんにちは" );
add( tck );
```

■：変数名　■：コメント　■：外部ファイル名やカスタマイズ可能なところ

これまでの説明では、Canvasという細かい設定までできるクラスを使っていたが、実はもっと簡単にプログラミングできるPanelクラスというのがある。これを使えば、文字を右から左へ流すといったHTMLでのマーキーのような表現をするのも、上のようなソースを書くだけだ。いままで見てきたCanvasと比べると、その単純さは歴然だろう。

では、なぜそれでもCanvasクラスについて解説しているかというと、実はPanelクラスは機種依存が激しく、作る側として使いづらいところがあるからだ。たとえば、D503iでは文字が16フォントに固定されてしまう、といったことがある。しかし、ソースコードが少ない行数で単純に書けるのはやはりうれしい。うまく組み合わせて使いたいところだ。簡単なので、ちょっと試しにPanelクラスを使ったプログラムを書いてみてはどうだろう。

メソッド：プログラミング言語において、あるオブジェクトが持っている関数や手続き。自分でプログラムを書かなくても、あらかじめ備わったメソッドを呼び出すだけでそのオブジェクトを簡単に制御できる。

今回注意すべき機種依存について

カウンターは時間を計算して動くため、どの機種でも同じように表現できると思うかもしれないが、残念ながら機種による違いが出てしまう。といっても、カウンター自身に問題があるのではなく、指定した時間からイベントを起こしてもそれを表現するまでの時間はCPUの性能に依存してしまうからだ。たとえば、画像を動かすというのは高い画像処理の能力を要求するために、それに関して劣る機種だとカウンターから出る命令に画面表示が追いつかないのである。特に今回は画像の消去と再描写を繰り返すことでアニメーション効果を出しているので、F503iではタイマーに追いついていけず、ほかの機種に比べて遅い動きになってしまう。

もし、全機種において同じような表現をしたければ、画像を小さくして画像処理にかかる負担を軽くしたり、タイマー間隔を広げることでCPUが画像処理を行う時間に余裕を持たせるなどして、一番遅い機種に合わせる形で調節しなければならない。

今月のオマケ！

3分で完成するオリジナルiアプリ

「自分の顔のパズルを作ろう」！

ここでは、ちょっとした変更だけで作れる自分だけのiアプリを紹介していく。
今月は好みの画像をパズルにできる「Puz3x3」だ。

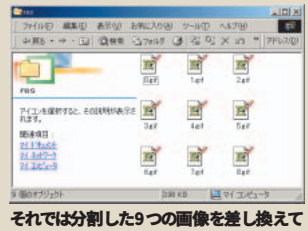
1 まずはダウンロード！

3×3パズルは、Jump internet.impress.co.jp/iappli/に置いてあるので、Puz3x3フォルダーごとダウンロードしてしまおう。

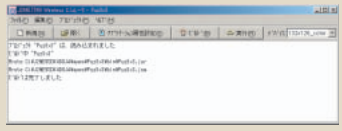
2 素材となる画像を分割しよう

Puz3x3は9つの画像の組み合わせでパズルを作る。そこでパズルにしたい画像を99×99ピクセルで4000KB以内で準備しておこう。そして3×3の9つに分割し、それぞれ左上から順番に0.gif、1.gifと順番に名前を付けていく。

3 画像を差し換える



それでは分割した9つの画像を差し換えてしまおう。



4 DoJaでコンパイルする

DoJaを使いコンパイルしよう。これで、新しい画像を使ったパズルに生まれ変わるぞ。なお、DoJaについて分からない人はDoJaのページJump www.nttdocomo.co.jp/i/java/tool.htmlを参照してほしい。DoJaについては本連載でも取り上げる予定だが、それまで待てない人は下で紹介している書籍を購入してもらいたい。

5 FTPでアップロード

自分のホームページエリアにアップロードしよう。その際、iモードからダウンロードするときに使うhtmlファイルを忘れずに。

6 完成！

オリジナルiアプリ募集中！

今月のオマケはうまく画像を分割できるかがポイントだ。ピクセル単位で切り分けよう。うまくできたら、オリジナルiアプリをアップロードしたサイトのURLを編集部宛てにメールしよう。もしできなくても、質問でも感想でもなんでもOKだ。応募作品はWe Love Internet Peopleでも紹介するので、ドシドシ送ってきてもらいたい。宛て先は次のとおり。 im-iappli@impress.co.jp

もっと知りたいキミにはコレ！

『今すぐできるiアプリプログラミング』
小社発行 本体価格2,400円
来月なんて待ってられない、もっとくわしくいろいろと知りたい！と思っている君にはコレがオススメ。この連載で取り上げているiアプリについても解説しているので、一緒に読んでいけばもっとよくわかるぞ。

DoJa（J2ME Wireless SDK for the DoJa）：NTTドコモから出されているiアプリ作成ツール。iアプリのソースコードをコンパイルする際に必要となる。

［インターネットマガジン　バックナンバーアーカイブ］ ご利用上の注意

このPDFファイルは、株式会社インプレスR&D（株式会社インプレスから分割）が1994年～2006年まで発行した月刊誌『インターネットマガジン』の誌面をPDF化し、「インターネットマガジン　バックナンバーアーカイブ」として以下のウェブサイト「All-in-One INTERNET magazine 2.0」で公開しているものです。

http://i.impressRD.jp/bn

このファイルをご利用いただくにあたり、下記の注意事項を必ずお読みください。

- 記載されている内容（技術解説、URL、団体・企業名、商品名、価格、プレゼント募集、アンケートなど）は発行当時のものです。
- 収録されている内容は著作権法上の保護を受けています。著作権はそれぞれの記事の著作者（執筆者、写真の撮影者、イラストの作成者、編集部など）が保持しています。
- 著作者から許諾が得られなかった著作物は収録されていない場合があります。
- このファイルやその内容を改変したり、商用を目的として再利用することはできません。あくまで個人や企業の非商用利用での閲覧、複製、送信に限られます。
- 収録されている内容を何らかの媒体に引用としてご利用する際は、出典として媒体名および月号、該当ページ番号、発行元（株式会社インプレスR&D）、コピーライトなどの情報をご明記ください。
- オリジナルの雑誌の発行時点では、株式会社インプレス R&D（当時は株式会社インプレス）と著作権者は内容が正確なものであるように最大限に努めましたが、すべての情報が完全に正確であることは保証できません。このファイルの内容に起因する直接的および間接的な損害に対して、一切の責任を負いません。お客様個人の責任においてご利用ください。

このファイルに関するお問い合わせ先
株式会社インプレスR&D
All-in-One INTERNET magazine 編集部
im-info@impress.co.jp