プチコン３号
ＱＳＰ完全マニュアル

もくじ

第1章　QSPとは何か

プチコン3号というのはスマイルブームから発売されている3DS用アプリでBASIC言語によって3DS単体でゲームやツールといったソフト開発が可能です。プチコン3号は従来のプチコンmkIIと比べて大幅に性能がアップしているためやろうと思えば16bitゲーム機並のものが作れるのですが、そこまでのものを作るのは非常に大変です。

そこで、お手軽に作れるようなプログラムとしてあえて使用する合計文字数を制限した1画面プログラムが個人的にはオススメです。プチコン3号の編集画面は46文字×29行となっているのでこれに収まるものが1画面プログラムとなります。（改行は画面内に含むかどうかは個人ルール、レギュレーションによって決まる）

プチコン3号のコミュニティでは1画面プログラムを「One Screen Program」略して「OSP」と呼ばれることが多いのですが、最近では「QSP」と呼ばれるさらに小さな1画面プログラムを作る人も増えています。「QSP」というのは「Quarter Screen Program」の略で直訳すると4分の1画面プログラムとなります。これはどういうものかというとプチコン3号 *ver.3.2.0*から新規に増えた命令**WIDTH**を使用すれば文字サイズを2段階に変えられるのですが、縦横2倍の文字サイズになる**WIDTH 16**を使用時の編集画面の1画面分に収まるプログラムのことです。この時、21文字×14行となり、デフォルトの1画面の4分の1よりもやや小さめの情報量となるわけですが、「概ね1/4」ということで**QSP**となっています。

QSPは上記のように使える文字数は最大で294文字（改行を画面内に含める場合には最大で293文字）となっています。実際の私の作例を挙げるとこんな感じとなっています。

※後述の **PETIT 100m QSP** の
プログラムリスト

「この文字数ではまともにプログラムなんて作れない」と考える人も多いかもしれません。しかし、実際に作っている人のプログラムを見ると一概にQSPではまともに作れないというよりもQSPだからこそお手軽に作れるというイメージが出てくるのではないかと思います。

実際シンプルなゲームやツールを作っても普通に作ったら「もう少し付け足したりして見栄えを良くした方が良い」とか「インデントや分かりやすい変数名を使うようにした方が良い」いう考えが先行してしまいプログラムリストの肥大化が進む一方ですが、QSPならばそれをいかにそぎ落とすかが重要になってくるためシンプルなゲームやツールであっても可読性が悪くても全く問題ありません。これは**QSPの場合はプログラムリストがWIDTH 16使用時の1画面に収まっているというのが何より優先される**ためです。

第2章　完成度が高いQSPを作るための基礎知識

QSPはお手軽に作りお手軽に発表というのも1つの醍醐味ですが、制限を最大限に活用してその範囲で作り込むというのも非常に楽しいです。最初から「QSPに収まる小さなプログラムを作るのが目的」というのあれば特に気にする必要はありませんがQSPという制限内で「いかに面白いゲームを作るか」「いかに実用的なツールを作るか」ということを考えるならばやはりそれなりに知っておくことがたくさんあります。そこでQSPを作るための基礎知識的なものをここでまとめることにしました。

一部のものはプチコン3号によるプログラミングを初めて間もないようなの初心者には難しいものや混乱を招くものもありますのでこれらをすべて活用するのはプチコン3号でのプログラミングに慣れてから行ってください。

完成度の高いQSPを作るための手順

(1)QSPで作れそうなくらいの小規模なプログラムのアイデアを用意する。
(2)実際に記述してWIDTH 16の1.5画面前後（エディタの行数ではなく実際に数えて20行前後）になればQSPにすることが可能になる。
(3)余分なスペースを詰めたり、リスト短縮を行ったり、優先度の低い処理を削ったりしてWIDTH 16の1画面に収まればQSPの完成！

完成度が高いものにするにはQSPであっても可能な限りは妥協をしないというのが重要です。そのためリスト短縮をしっかり行い自分が**「これは入れたい」という要素は確実に入れる**ようにしてどうしても**QSPに収まらないという場合には優先順位の低いものから順番に削っていく**必要があります。（優先度が低いものを削ることを前提に考えれば当初は「20行前後」に拘らずそれよりも多少大きなものでも問題ない）

そのため求められるのは**自分が作りたいものに入れたい要素の優先順位を明確にする**ということと**リスト短縮テクニックを身につける**ということです。ゲームバランス調整や画面レイアウトなどはQSPならではの部分もありますが、普通に作るプログラムとそれほど変わりません。（QSPだから画面やシステムをシンプルにすることがあってもゲームバランスを自分が作りたいものから変えてしまうというのは個人的には避けたいところ）

それでは、まずはQSPを作るのに必要ないリスト短縮テクニックの中から主なものを書いていこうと思います。

QSP作りに必要なリスト短縮テクニック

(1)省略可能なものは省略

プチコン3号においてコロン（":"）やスペース（" "）が省略できる場面は非常に多いです。命令、関数や定数とかを記述する場合に末尾が数字や閉じカッコが付いている場合はほぼ省略可能です。定数においてはデフォルトでピンク色になるため定数がピンクではなく白色になってしまうような場面のみコロンやスペースを付ければ良いだけです。

（例1）
```
A=12:B=45  → ○ A=12B=45   （コロンの省略が可能）
A=X1:B=Y1  → × A=X1B=Y1   （コロンの省略は不可）
```
※X1というのは定数ではなく変数であるため

（例2）
```
X=SIN(A)*B:Y=COS(A)*B
```

このままだとコロンは省略できないけど乗算は順番を入れ替えても問題ないためSIN(A)*BをB*SIN(A)に変えてみます。

```
X=B*SIN(A)Y=B*COS(A)   （コロンの省略が可能）
```

また、演算優先順位を把握し、必要最小限のカッコのみ残すことで不要なカッコを省略することもできます。

命令によっては引数を省略可能な場合があります。

```
VSYNC      （VSYNC 1から「1」を省略したもの）
BEEP       （BEEP 0から「0」を省略したもの）
LOCATE,Y   （前回表示したX座標と同じ座標に表示させる場合は省略可能）
```

他にも省略できるものはたくさんあるので積極的に活用していきましょう。

(2)論理式の使用による短縮

A>1などの条件式においては**成立していれば「1」、不成立の時には「0」**の値を取ります。この条件式のことを論理式とも言いますがこれを活用すれば短縮できることが多いです。
Aの値が10以上の時にBの値は3、Aの値が10未満の時にはBの値は1になるようにしたいという場合には普通にIF文で記述すれば次のようになります。

```
IF A>=10 THEN B=3 ELSE B=1
```

しかし、論理式で記述すればこのようにできます。

```
B=(A>=10)*2+1
```

短くなるだけではなくIF文と違いこのBの代入式の後にも別の処理をマルチステートメントで記述することができます。マルチステートメントはコロンで区切って1つの行に複数の命令などを記述するものですが、上記の例では定数「1」が末尾なのでその区切るためのコロンも省略可能です。

この**論理式は有効活用できる機会が非常に多いので必須テクニック**といっても過言ではありません。

(3)条件判断の短縮

IF文においては条件式の値が「0以外の場合はTHEN以下を実行する」というのを覚えておけばリスト短縮ができる場面が非常に多くなります。

```
IF A!=0 THEN BEEP  →  IF A THEN BEEP
```

この場合は「**!=0**」は省略が可能です。Aが負数にならないのが分かっているならば**A>0**の「**>0**」も省略が可能となります。

```
IF A==0 THEN BEEP  →  IF !A THEN BEEP
```

IF文においては条件式が**論理反転を示す演算子"!"は「0を1」「1を0」**にできます。(正確には「0以外を0」にする)

あと(1)では書きませんでしたが変数名で使えない変数や命令の後に(変数名としては使えない)記号が続く場合にはスペースが省略可能です。したがって、「IF」と「!」の間のスペースは省略可能になっています。

```
IF A!=0 && B!=0 THEN BEEP
```

これはIF A&&B THEN BEEPと短縮することが可能になるのは分かると思いますが、実はIF A*B THEN BEEPとさらに1文字短縮が可能です。

THEN以下を実行するか否かは0以外か否かというのはすでに書いた通りですが、AとBの両方もしくは片方が0の時はA*Bの値は0になり式の値は0になるためTHEN以下が実行されないのでこれで全く問題ないのです。このようにかつを「*」、またはを「+」で表現した方が短縮できる機会もあるというわけです。(単純な置き換えはできないので使用する場合にはよく考えて使う)

(4)ループ処理の短縮

プチコンにおいてはループが可能な命令がいくつも用意されています。QSPを作る場合には次のように使い分けると良いでしょう。

(a) 単純に繰り返したい場合

```
@L   (中略)   GOTO@L
```

※@Lでなくても1文字ラベルならば良い。GOTOと@Lの間のスペースは省略可能

(b) 回数を指定、もしくはインクリメント処理が必要な場合

FOR ～ NEXT を使う

FOR～NEXTが最短になるのは刻み値が1の場合のみです。(**STEP**を省略可能な場合のみ)

(c) 指定条件でループを抜けたい場合

WHILE ～ WEND を使う

WHILE～WENDは**最初から条件が不成立の場合はループを1回も実行せず**に**WEND**の次の処理へと移りますが、それでは困る場合には最初は成立する値を設定しておくか、条件式を変えるか、***REPEAT～UNTIL***を使用してください。これは代入と***REPEAT～UNTIL***の短い方を使用すれば良いです。

(5)表示処理の短縮

超基本としてはコンソールの文字表示は**PRINT**の代わりに「**?**」で代用ができるということは覚えておきましょう。
ゲームにおいてタイムなどを表示の際にネックになるのは桁数が減る場合です。その場合には減った部分はそのまま残ってしまうのでその都度消去処理が必要です。ループ内に**CLS**があれば問題ないのですが、そうでない場合には前回表示した文字を消す必要があり、そのためにはLOCATEで指定するので無駄に長くなるという問題があります。

```
?"TIME";T;"   "
```

このように、表示の後にスペースを入れれば減った分の文字をスペースで自動的に消去が可能ですが次のようにすればさらに短縮が可能です。

```
?"TIME";T,
```

このようにタブ表示機能を使えば不要な文字は簡単に消去可能です。

あと、ゲーム内でグラフィックやスプライトやBGを使わない場合にはACLSは省略可能です。(前回プレイしたゲームにそれらが使われていたり、GRPページが変更されていたりということを考えると個人的にはACLSをよほどのことがない限りは入れたいと考えているけど)
リスト内にWIDTH 16を入れているプログラムの場合は***WIDTH 16***で**CLS**の代用ができるので**CLS**が省略可能です。(速度が遅いので余力があるプログラムに限られるけど)

LOCATEとPRINTを多数使用するような場合には自作命令を作ることでリスト短縮ができます。

```
DEF P X,Y,A$:LOCATE X,Y?A$END
```

このような自作命令を使えば、LOCATE 12,3:PRINT "ABC"をP X,Y,"ABC"とすることができます。
LOCATEを多用する場合に限らず、**特定の処理を何度も行う場合にはこのように1文字命令の自作関数(命令)を作ることはリスト短縮においてかなり役立つ**ことがあります。(このような処理は旧来のBASICだとGOSUBが使われることが多いけどQSPにおけるリスト短縮ではGOSUBはほとん

ど役に立たない)

リスト短縮テクニックを使用した場合はプログラムリストの可読性を大きく損ねることがありますが、**完成度の高いQSPにしたい場合には可読性が犠牲**になります。この辺は一般的なプログラミングの手法とは大きく異なる部分といえるでしょう。(普通は可読性を犠牲にしてまで短くする必要性はないので**完成度を高めるために可読性を下げるのは文字数に制限があるプログラムを作る場合のみ**)

リスト短縮テクニック以外のものは次の章からの実践編に具体例付きで書いています。上記に記載されていないリスト短縮テクニックを使用している場合もそれも合わせて書いていきます。

第3章　実践編1　100m走ゲームを作る

「100m走ゲーム」とは何かというとボタン連打を行い画面上のキャラを走らせ100m走としてのタイムを競うゲームです。シンプルな内容ながらハイパーオリンピックなどたくさんゲーム化されている結構メジャーなジャンルです。内容がシンプルであるため昔の遅い8bitPCやポケコンのBASICでも多くのプログラムが作られました。「100m走ゲーム」は私も過去に数100作品作り作ったゲームはベーマガ等の雑誌にも多数掲載されました。

「100m走ゲーム」は単にボタン連打でキャラをゴールまで走らせるだけのシンプルなゲームとはいえそれでも作る人によってその手法が変わってきます。今回はQSPということで見た目に拘ったものにするのは難しいですが、ゲーム内容、ゲームバランスは「QSPだから仕方がない」という妥協をせずに作ろうと思います。

さて、100m走ゲームはQSPということで表示はシンプルにスクロール無しで画面幅が400ドット分なのでスタートから300ドット進んだらゴールという設定にしようと思います。それを元にコースをＧＢＯＸで描画します。

あとプレイ方法は1つのボタンを連打ではなくAボタンとBボタンの交互連打でキャラを走らせようと思います。

これによってできたのがこの100m走ゲームの試作品です。

《 100m走ゲーム 試作品 プログラムリスト》

```
ACLS:WIDTH 16
X=37:GCLS #GREEN
GBOX 50,69,349,99,-1
SPSET 0,716
SPOFS 0,X,75
?"READY";:WAIT 99?"GO!"
BGMPLAY 41
SPANIM 0,3,4,716,4,717,4,718,4,719,0
WHILE X<337
 A=B:B=BUTTON()
 IF A==16 && B==32 THEN X=X+3
 IF A==32 && B==16 THEN X=X+3
```

```
 SPOFS 0,X,75
 T=T+1/60
 LOCATE 8,9?FORMAT$("%5.2F",T)
 VSYNC
WEND
BGMPLAY 5?"GOAL"
```

これで、WIDTH 16の場合には画面上では23行分になっています。

実行してみるとメイドさんの動きが若干カクカクしているのが気になるかもしれません。
また、連射が速い人ならば7秒台、8秒台という超人的なタイムが出せる反面で遅い人だと10秒台後半～20秒のタイムになってしまいます。これは私が100m走ゲームを作る時に気にしていることですが、**連射が普通（秒間10回程度）で12秒前後、やや速め（秒間12回程度）で11秒前後、速め（秒間15回程度）で10秒前後のタイムが出せるようにしているのでQSPであってもこれは妥協せずに作ろう**と思います。（これは実際の100m走競技に近いレベルのタイムが出るようにするため）

実はカクカクしないようにするというのとタイムのバランス調整は速度のパラメータを導入するだけで一度に解決できるのです。というのも「カクカクする」というのは正しくボタンを押せている時のみ横移動を行いそれ以外は移動が停止しているため起こっているからです。
ただし、速度の場合は上限と下限の判定を行う必要があり、現状でQSPとしてはギリギリのサイズ（むしろ、ここから何かを削らないと厳しい）になっているためかなりのリスト短縮が必要となります。
あとゲームオーバー時にはすぐに終了してしまうため**TOP MENU**でファイルを選んで実行時にはゴールした瞬間にメニュー画面に戻るため「プログラムを作る」からしか実行できないという問題があります。これも改善しようと思います。

大幅にリスト短縮ができるのはキー入力判定部分でこれはIF A==B THEN A=48-Aとすることで1行に収まります。正しく押された時には加速処理を行いそうでない場合には減速処理を行えばいいです。その時の加速と減速の値は上記のような連射速度とタイムの関係になるような値にすれば良いです。
連射が極端に速い場合も一定のタイムになる簡単にするためには加速度に上限を付ければ良いです。これは現状の速度が速ければ速いほど加速度を減らすというものです。速度はループ1回ごとに同じ割合で減少させていけば正しく押した時は速度アップ、正しく押せてない時はわずかな速度ダウン、速度が0になっているかどうかを判断する処理が簡単に記述することができます。
ここでは、加速度はP=(1-S)/12、速度はS=P+S*0.98としました。つまり、ループ1回ごとに速度が2%ダウンするけど**P**の値はそれよりも大きくなるため押せば速度アップとなるわけです。**1-S**というのは速度が最大時でも1を超えないためで12がバランス調整用の係数です。

ゲームーオーバー時にすぐに終了してしまうのを避けるには最も簡単なのは**WAIT**を入れることです。またAボタンを押すまで待つという処理にしたければLINPUT A$やDIALOG""でもOKです。ただし、このゲームでは**A**ボタンをゲームに使っているため**A**ボタンを押すまで待つ処理というのはWAITとの併用が必要不可欠になってきます。
1回のプレイ時間が短いゲームの場合はリトライ機能（終了ではなくもう1度プレイ）を付けるというのがベターですが、QSPの場合はそれも困難であり、リトライよりも優先度の高い処理が多いため難しければ無理にリトライを付けなくても良いと思います。（リトライを付けることでゲーム本編の処理を妥協しては本末転倒）

リスト短縮はボタン入力処理を1行にすることで大幅にリスト短縮ができましたが、それでも上記のような速度、加速度処理やゲーム終了時にWAITを追加すればさらなるリスト短縮が必須となります。

そこでタイム表示の短縮をしてみます。タイム表示で目立つものといえばFORMAT$でしょう。これは小数第2位までを指定することで無駄に小数第8位まで表示されるのを避けることが可能になっています。

実はＦＯＲＭＡＴ＄を使わなくてもTの値を表示する際に小数第2までに丸めてやれば解決です。これは**100倍して整数化を行い100で割れば良い**です。

？ＦＯＲＭＡＴ＄（"％５．２Ｆ"，Ｔ）は？ＦＬＯＯＲ（Ｔ＊１００）／１００，とすることができます。とはいえ、あまり短縮にはなってないですね。

しかし、Ｔ＊１００というのは省略できます。

これは、Ｔ＝Ｔ＋１／６０をＴ＝Ｔ＋１．６６にすればいいです。このTの値は厳格な数字である必要性はないのでこれで問題ありません。（場合によっては1.66という数字は1.6や1.7にしてもOK）

あと整数化ですがこれはＦＬＯＯＲを使う必要はありません。プチコン3号は整数演算が行われる演算子では自動的に整数化が行われるためシフト演算を使いＴ＞＞０でＴの値を整数化が可能です。（負数の場合はＦＬＯＯＲと挙動が変わるので注意）

定数リテラルを使いＧＣＬＳ ＃ＧＲＥＥＮで背景に緑を表示しましたが、これはＲＧＢ（０，１２８，０）の緑に拘る必要はないため緑っぽい色を表示するため ＧＣＬＳ－１５５Ｅ５としました。（-155E5は-15500000のことだけどプチコン3号ではRGB関数を使わずとも32bit論理色コードでダイレクトに指定可能なので自分のイメージに近い色を数字で指定してやれば良い）

あとメインループ内にあったＶＳＹＮＣはＷＡＩＴに変更しています。これは60fpsで動作しているプログラムの場合はVSYNC 1はWAIT 1に置き換えることができるためです。（VSYNC 2とWAIT 2は挙動が異なる）

以上を元に作ったのが完成版のゲームである**「PETIT 100m QSP」**です。

QSPを作る場合には単純に短縮するというだけではダメで処理の並べ替えを行い行末をできるだけ空かないようにするという工夫が必要になります。この「PETIT 100m QSP」は改行コードを含めてぴったりWIDTH 16の1画面に収まっていて無駄な文字が一文字もありません。つまり、ここまでリスト短縮などを行いようやく当初考えていた要素をすべて詰め込むことができたというわけなのでリスト短縮テクニックの重要性が分かると思います。

第4章　実践編2　音当てゲームを作る

今度は、音当てゲームを作ってみます。これはC、D、E、F、G、A、B（ドレミファソラシド）をランダムに演奏してそれをプレイヤーに答えさせるというものです。正答すればレベルが上がり、レベルが高くなればなるほど音の数が大きくなることで難しくします。

実はこのプログラムはすごく簡単です。

《　音当てゲームプログラムリスト　》

```
WHILE !C
 L=L+1
 A$=""FOR I=1TO L:A$=A$+CHR$(RND(7)+65):NEXT
 ?"Lv ";L
 BGMPLAY"L"+STR$(L)+A$
 WAIT 120
 INPUT"答え";B$?"正答 ➡";A$
 C=A$!=B$
WEND
?"GAME OVER"
```

プチコン3号は実行するごとに変数が初期化されるためCの値はスタート時には0になっています。そのため正答時にはCの値を1にするというのであればＷＨＩＬＥ～ＷＥＮＤを使うならばその前にＣ＝１を入れるかＲＥＰＥＡＴ～ＵＮＴＩＬにする必要があります。

しかし、そんなことをしなくてもミスの時にＣの値が1になるようにしておけばＷＨＩＬＥ　！Ｃとするだけで問題ないのです。

あと音の数が多くなる場合はＢＧＭＰＬＡＹにおいて**L値（デフォルト音長）をレベルの値（鳴らす音の数）で設定するだけで2秒固定**で簡単に演奏ができます。これはデフォルトでは**T120**になっていて全音符1つあたり2秒間演奏が可能であるためです。

何もしなくても余裕でQSPに収まっているプログラムですがこのプログラムも気になる点があるためそれを改善していきます。

ゲームバランス調整のためミスした場合の一発ゲームオーバーはやめて2回連続でミスした場合のみゲームオーバーとします。1回ミスをした時に何のペナルティも無いのは問題なのでペナルティとしては普通は「1回正答したらレベルアップ」のところを「ミスをした後は2回連続で正答でレベルアップ」とします。

これによって、その人の実力にあったレベルを何回か繰り返すことになるため「すぐにゲームオーバーになってつまらない」とか「1回のゲーム時間が長すぎる」とかいう問題も解決できます。

ゲームバランスの調整を行っても余裕があるのでQSPでできる範囲内で演出などに拘っていきたいと思います。（少しでもいろいろな演出を加えるためリスト短縮も可能な限り行っていく）

演出の方向性をタイトル名で記すためにタイトル名は「音当てゲーム」ではなく「音姫～サウンドプリンセス～」としました。

そのためデフォルトの姫様（というか女王様）のスプライトを使用します。このキャラを拡大表示して流れてくる音（♪）を聴いているようなイメージの画面にしたいと思います。

正解、不正解、ゲームーバー時は音を変えていきます。これは正解の場合は「やったね」、不正解の場合は「いやん」、ゲームーバー時は「うわ～ん」と号泣させようと思います。セリフはＢＥＥＰの音

を鳴らすだけですが、号泣は音姫らしく（？）30秒間水が流れる音を鳴らします。

以上の改善案を元に完成したのが、この**「音姫～サウンドプリンセス～ QSP」**です。

上記の演出を加えるだけでQSPに収めるためにはかなりのリスト短縮が必要になりました。

例えば、スプライトの表示座標に関しては１２８というのを指定すれば3文字必要ですが、定数リテラルで＃Ｙとすれば2文字で済むためこういう本来は想定していないであろう方法で定数リテラルを活用しています。

水流音ですが、ＢＥＥＰ ８５で簡単に出せるもののこれだと2秒くらいしか鳴りません。そこで隠し音色として用意されている＠３８６を使用します。30秒というのはＴ２の四分音符なのでＢＧＭＰＬＡＹ "＠３８６Ｔ２Ｃ" とすれば30秒間水流音を流すことができます。

第5章　実践編3　お絵かきソフトを作る

QSPでお絵かきソフトとなると無茶なような気もしますが、プチコンで絵を描くプログラムというのは簡単に作れます。

《超簡易お絵かきプログラム　リスト》

```
XSCREEN 2
DISPLAY 1
GPAGE 1,1
WHILE 1
 BX=X:BY=Y
 TOUCH OUT T,X,Y
 IF T>1 THEN GLINE BX,BY,X,Y
 VSYNC
WEND
```

XSCREEN 2（もしくはXSCREEN 3）で上下画面が使えるようにし、DISPLAY 1で下画面が操作できるようにするだけです。GPAGE 1，1は最初にACLSを入れておけば省略が可能です。(ACLSを実行すれば自動的に下画面はGRP 1になるけどこれは事前に別のソフトを使っていてGRPページが変更されている場合も正常に動作するようにするためには必要となる)

あとは、**タッチしているのが2フレーム以上ならば前回タッチした座標と今回タッチした座標を線で繋ぐだけ**です。

プチコン3号は仕様上、下画面の上下左右の5ドット分の外枠は正常に座標判定ができないため隅の方まで描きたいならばキャンバスがスクロールするお絵かきソフトを作る必要がありますが、QSPでは無理なのでこれは考えないことにします。それでもラクガキ程度ならば特に困ることはないでしょう。

このプログラムではあらかじめGCOLORで指定した色（デフォルトでは白色）を使い幅1ドットの線で描画が可能ですが、線の太さも色も変えることができず、消しゴム機能もありません。あとファイルのセーブ、ロード機能もありません。そのためいくらQSPだからシンプルなものになってしまうとはいえこの辺を改善したいと思います。

要するにQSPの短さを活かし**ラクガキに特化したシンプルかつ直感的に使用できるようなものを作る**ということです。こうすることで、単なる低機能ソフトにはならずすでに多数発表されている高機能、多機能なお絵かきソフトとの棲み分けが可能な「ラクガキ専用のお絵かきソフト」としてQSPであっても十分に実用となるソフトになります。

まず、線の太さを変えるにはどうするかというとタッチした座標に塗りつぶし円を表示するだけです。しかし、それでは問題点が2つあります。

線の太さを自由に設定できるプログラムを作る場合の問題点

(1)GCIRCLEとGPAINTを普通に使って塗りつぶし円を描くと正しく描画されない

(2)ペンを速く動かすと線が途切れてしまう

問題点(1)がなぜ起きるかとうとGCIRCLEで円を描いた時点で円内にすでに何か描画されていた場合にはその部分が境界になって正常な塗りつぶしができないため発生しています。実は塗りつぶし円に

はたくさんのアルゴリズムがありそれらを1つ実装すれば解決できます。
簡単で短いものだと私が昔ポケコンで作った三平方の定理を使いGLINEで円をライン単位で描画していくというものです。

```
DEF GCFILL X,Y,R,C
 VAR A,I
 WHILE I<R
  A=SQR(R*R-I*I)
  GLINE X+I,Y+A,X+I,Y+A,C
  GLINE X-I,Y+A,X-I,Y+A,C
  I=I+1
 WEND
END
```

しかし、この塗りつぶし円の描画アルゴリズムとしてはコンパクトなものに位置するそのプログラムでさえも、QSPに導入するにはやや長目なので別のアルゴリズムを考えてみます。

まずは仮の境界色を設定してその色でGCIRCLEを描画してその仮の境界色を基準にGPAINTで塗りつぶします。そのあと正規の色でGCIRCLEで描画すれば正しく塗りつぶし円が描画可能です。
しかし、その仮の境界色はその絵で絶対に使用しない色に設定しなくてはなりません。
そこで考案したのが描画色8です。これはARGBで表すとRGB(0,0,0,8)です。これは、透明色になるため普通はGRPでは使わないし、透明色の中で0以外の数字で最も文字数が少ないのは「8」です。
「8」でなくても「1」や「2」でもいいと思うかもしれないですが、**プチコン3号のGRPは表示の際に色コードは8の倍数に丸められるため**1や2で設定すると0扱いになります。これはGCLS 1:?GSPOIT(0,0)だと0になりますが、GCLS 8:?GSPOIT(0,0)は8になることで簡単に確認が可能です。(GCLS 9でも8になる)

```
DEF GCFILL X,Y,R,C
 IF R<1 THEN
  GPSET X,Y,C
 ELSE
  GCRICLE X,Y,R,8
  GPAINT X,Y,C
  GCRICLE X,Y,R,C
END
```

この塗りつぶし円描画プログラムは中心座標がX、Yのどちらかが0～511の範囲から外れた場合には正常に描画はできませんが、その問題が発生する座標にはタッチそのものができないためお絵かきプログラムに使うならばこれで何ら問題はありません。
Rの値が1より小さい場合はGPAINTで塗りつぶせないためその例外処理を行っています。そのためリスト短縮の面では効果は薄いですが、円の描画のキレイさと後述のカーソル表示機能によって十分なメリットはあるでしょう。

問題点(2)はTOUCH OUTで取得した座標をGPSETで描画して描いたら線が途切れ途切れになるけど前回タッチした座標と現在の座標をGLINEで繋げば途切れない線として描画されるのと同じことを(1)で考えた塗りつぶし円に対しても行えば良いだけです。
これは前回タッチした座標と今回タッチした座標を1ドットずつ座標を変えた塗りつぶし円で描画すれば解決できます。つまり、X座標、Y座標が何ドット動いているかを調べてその多い方のループ回数

実行すれば良いだけです。

問題点(1)、(2)を解決したら新しい問題点としてその線の太さをどうやってユーザーに選択させるかというのが重要になってきます。

QSPではメニュー画面から選択というのはサイズ的に困難だしそんな普通の方法ではすでにある高性能なお絵かきプログラムではなくQSPを使うメリットがないためここでは1ボタン選択を考えていきます。つまり、機能が少ないからこそボタンの割り当てが余るため太さ1ドット、2ドット、4ドット、8ドット専用のボタンというのを用意することができるわけです。

ここでは、最も多用するであろう太さ1ドットの場合は何も押さない場合、太さ2ドット（正しくは円の半径で言えば1ドットなので太さは3ドット分になる）が十字ボタン上、半径2ドットが十字ボタン下という感じで割り当てますこうすればＢＵＴＴＯＮ（）関数の値をそのまま使えるのでリスト短縮に非常に有用です。

次は消しゴム機能ですが、これはキャンバスの元の色で描画すれば良いです。つまり、何かボタンを押すことで描画色とキャンバスの色の2色を切り替えればいいわけです。ただし、ボタンの割り当ての煩雑化を招くため描画色は1色で固定（他の色を使いたい場合はプログラムを書き換えで対応）としてます。

ファイル操作は描画に比べたら優先度は低いので必要最小限のもののみ搭載させます。

というわけで、完成したのがこの「PETIT PAINT QSP」です。

「何も押さない」、十字キーの「上」「下」「左」「右」で線の太さが「極細」「細」「中」「太」「極太」と変化します。（ABYXボタンを押すことで超極太にしたり、ボタンの同時押しで太さを微調整したりも可能）

消しゴムはメニュー画面から選ぶということは不要でLボタンもしくはRボタンを押しながら描くだけでいいです。多機能ソフトでは普通となるメニュー表示などによって選んだりするということを行う必要があり常にキャンバスを見ながら描くということはできませんが、このソフトではキャンバスのみを見ながら描くことが可能になります。

ちなみに線の太さに応じたカーソル表示が行われるため線の太さがどれくらいか分からないという心配もないです。これは太い消しゴムを使用する場合には非常に有用でどの範囲内が消しゴムによって消えるのかが一目瞭然です。

ファイル関係は一時ファイルへのセーブのみサポートしています。Yボタンを単独で押すことでファイル名"ＧＲＰ１：@"でセーブが可能です。これはMiiverseへの投稿するときにエラーでプチコン3号が落ちて描いた絵が消えてしまったときのための保険代わりにするとか、復元ポイントにするとかいう使い方を想定しています。

一時保存ではなくちゃんと保存したい場合はＲＥＮＡＭＥ "ＧＲＰ１：@"，"ＧＲＰ１：（ファイル名）" として異なるファイル名に変更してください。ファイルのロードは1行のコメントアウトしている部分を外すことで可能です。一時ファイル以外からのロードを行いたい場合はこのLOADの部分のファイル名を変更してください。

線の色はデフォルトでは赤になっています。これを変えたい場合は2行目の定数リテラルの#REDを変えてください。定数リテラルではなくRGBで指定すればプチコン3号で使える色をどれでも使うことができます。(1度に描画に使える色は1色のみ)

なお、高機能お絵かきソフトによく搭載されている擬似筆圧搭載バージョンも用意しました。こちらは「PETIT PAINT QSP ver.B」です。

```
00001 Q=16ACLS:XSCREEN 2DIS
      PLAY 1GCLS-1@L VSYNC:
      U=X:V=Y:B=BUTTON()TOU
      CH OUT T,X,Y:C=B AND
      768P=(P+(P<Q))*!!T:R=
      (B AND 255)*P/Q:C=-!!
      C+!C*#RED:IF T>1THEN
      IF R<1THEN GLINE U,V,
      X,Y,C ELSE FOR I=1TO
      64M=U+I*(X-U)/64N=V+I
      *(Y-V)/64GCIRCLE M,N,
      R,8GPAINT M,N,C,8GCIR
      CLE M,N,R,C:NEXT
00002 GOTO@L 'SAVE"GRP1:@"
      0  2  4  6  8  10 12 14 16 18 20
```

1行目のＱ＝１６の値を変えることで筆圧の効き具合が変わってきます。原理は単純でタッチしている時間で線の太さを変えているだけです。

なお、この筆圧機能を入れる代わりにセーブ機能とカーソル表示機能は省きました。

個人的には筆圧は優先順位は低めなのでですが、「擬似筆圧ならばQSPに収めることも可能」ということを言うために作ってみただけなので（ラクガキを実際に描く場合の）実用性は筆圧に対応していない通常版の方が上でしょう。

第6章　まとめ

今回、QSPの基礎知識と実践テクニックをいろいろと書いてみましたが、やはり、QSPは難しく考えずにお手軽に作ってみることが一番だと思います。
まずは、「ごく小さなプログラム」としてQSPを作り、ある程度プログラムを作ることに慣れた人が制限付きで作ってみたいという場合にはQSPが制限付きプログラムとしては最もお手軽になります。「QSP」というのを機能が低いツール、画面が寂しく単純なゲームの免罪符として使用するというのも問題はないと思います。（プログラムサイズが大きくなれば機能を増やすこと自体は難しくないため「シンプル」をウリにするのはありだと思う）
見ての通りQSPのサイズでも様々なジャンルのプログラムを作ることは可能です。

QSPやOSP（普通の1画面プログラム）を作ることでリスト短縮テクニックが有効活用できるのですが、QSPはOSPよりも文字サイズが少ない分だけお手軽に作ることができます。作るプログラムの難易度にもよりますが、QSPに収まる300文字程度のプログラムだとアイデアさえあれば10分とか20分とかで作ることができると思います。（「改行コードはすべて画面内に含める」というレギュレーションの場合は改行コードを除いて最大293文字記述が可能なので概ね300文字程度が限界と思っておけば良い）
もっとも、そこからリスト短縮をするのが大変で私も1作品当たり数時間かかってしまっているのですがリスト短縮にそこまで拘らなければ1時間足らずで1作品を完成させることができます。とはいえ、どうしても入れたい機能があるにも関わらず、あと数文字足らないから入れられないという場合にリスト短縮などによって可能になれば入りそうという状況ならば個人的には無理に完成を急がず入れたい機能を入れてから発表という形を取っていますが、この辺は作る人の自由でいいと思います。

しかし、「少しでも完成度を高めたい」というのならばQSPであっても制限のない普通の作品と同じく妥協をしないということが求められてきます。この本を読めばそれがどういうことを意味しているのかが分かるのではないかと思います。

付録

今回の3作品（＋１）の公開キー　一覧

◎PETIT 100m QSP
公開キー ＡＢＷＸＥ３４Ｅ　ファイル名 １ＧＱ＿１００Ｍ
◎音姫～サウンドプリンセス～ QSP
公開キー ４３ＥＫＱＪＡＹ　ファイル名 １ＧＱ＿ＯＴＯ
◎PETIT PAINT QSP
公開キー ＥＤＷＸ３３８Ｖ　ファイル名 １ＧＱ＿ＰＮＴ
◎PETIT PAINT QSP ver.B
公開キー Ｋ２５Ｙ４３Ｋ４　ファイル名 １ＧＱ＿ＰＮＴＢ

※公開キーやファイル名は変更される場合があります。
変更した場合の公開キーや誤植などがあった場合は下記のサポート用URLに
記載しますのでご覧になってください。

http://ochameclub.web.fc2.jp/CLUB/c88.htm

プチコン3号 QSP完全マニュアル

2015年8月16日発行
制作・発行　おちゃめくらぶ
発行人　　御茶目菜子（おちゃめ）

おちゃめくらぶ　Webサイト　http://ochameclub.web.fc2.com/
おちゃめ　twitter　https://twitter.com/ochame_nako
おちゃめ　pixiv　http://pixiv.me/ochame_nako

おちゃめくらぶ 2015