

# 春休みプログラムセミナー 「プチコンでプログラム」

2012.3.20 上田市マルチメディア情報センター

# セミナー方針

## \* 想定している受講者像

- \* プログラムのことは知らないけど、ちょっと興味がある
- \* BASIC（ベーシック）は見たことも触ったことも無い
- \* ゲーム専用機やスマホや携帯などでゲームはする
- \* DSは知っているし触ったこともある

## \* 説明方針

- \* 派手な3Dやゲームっぽいキャラクタは使わずシンプルな表現にとどめます
- \* 難しいことは詳しく説明しません（理論とか詳細仕様など・・・）
- \* 資料から文字を入力してプログラムを動かすことを最優先とします
- \* 動いた後に改造すると面白い要素について説明します
- \* 将来のために三角関数等のあえて難しい処理も入れておきます

# セミナー進行予定

- \* 10:10 講師紹介
- \* 10:15 コンピュータで遊ぶ
- \* 10:30 お絵かきツールの開発
- \* 12:00 お昼休み
- \* 13:00 ゲームの開発
- \* 15:00 休憩
- \* 15:10 プログラマーになるには
- \* 16:00 終了



プチコンは3Dに対応していないため今回のセミナーでは3D表現のゲームは作りません

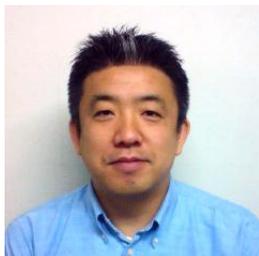


このゲームはプチコンで作られたキャラクターを使ったゲームですが、今回のセミナーではキャラクターを少しだけ使います。

```
MML PLAYER v1.0
1) MML プレイヤー 4 ショウキカ
2) BGM 1 2 3 4
0 : @16 C D E F G H B < C >
1 : @18 V 0 0 0 P P L N 2 C R R R G R G R
3 : @30 P 3 2 2 G F C C G F C C C H
4 : @19 L 1 6 C C C C C C C C C C C C C C
5 : @31 L 1 6 C C C C C C C C C C C C C C
6 : @53 3 L 1 6 R R R R R R R R R R R R C C R
7 : @30 @ 5 4 L 1 6 R R R R R R R R R R C C R
8 : @48 L 1 6 R R R R R R R R R R R R C C
3) BGM I ソウチュウ
```

コンピュータに命令することを実感してもらうため、今回のセミナーでは基本的に文字中心で進める方針です。

# 講師紹介



**20年以上前から札幌でゲームを作り続けています。  
北海道でゲームを作る若者を増やすために活動中。**

名前	小林貴樹（こばやしたかき）
所属	株式会社スマイルブーム
特技	面白い企画発案、簡単なプログラム作成
その他	CEDECアドバイザリーボードメンバー Microsoft MVP2011
関連作品	うっでいぽこ、ヴォルガード2、パワフル麻雀 俺の料理、ガチャろく、だれでもアソビ大全 アクションゲームツクール、プチコン

**CEDEC2011**  
Computer Entertainment Developers Conference



# コンピュータで遊ぶ

ニンテンドーDSもコンピュータです。

1

# 1-1 コンピュータとBASICとプチコン

## \* コンピュータは生活に溶け込んでいます

- \* ゲーム専用機、携帯電話、スマートフォン
- \* 家電製品、車、銀行のATMなど・・・

## \* コンピュータの役割

- \* 数値の記憶と計算、条件比較と分岐
- \* 入力操作受付、画面への表示
- \* 0と1だけの機械語という命令で動く

## \* BASICの役割

- \* 機械語を人間用にわかりやすくした命令言語

## \* プチコン

- \* ニンテンドーDSi上で動くBASIC言語



# 1-2 コンピュータ (DSi) へ命令してみる

## \* 実行モード



- \* BASIC命令を直接入力することができる画面
- \* 実行モードからの入力を試してみよう
  1. 命令1-2Aの内容を下画面のキーボードから入力
  2. 入力が終わったら「Enter」キーを押す
  3. 上画面に円が描画される
  4. 続けて、命令1-2Bと命令1-2Cも試してみましよう

命令1-2A : 画面に赤い円を描く

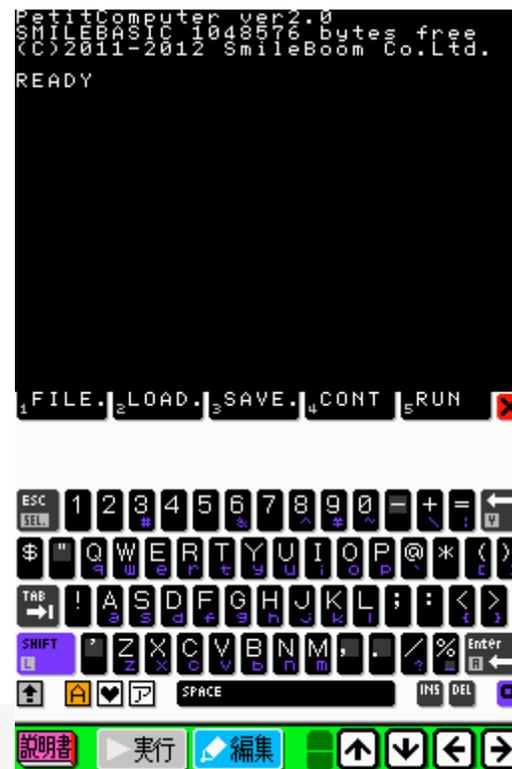
```
GCIRCLE 128, 96, 80, 2 ↵
```

命令1-2B : 円の中をグレーで塗る

```
GPAINT 128, 96, 1 ↵
```

命令1-2C : 図形画面を消す

```
GCLS ↵
```



全ての入力が終わったら「Enter」キーを押すと命令がコンピュータに伝わります。

# 1-3 一度にたくさん命令する

## \* 編集モード



- \* コンピュータへの命令を並べて書く画面
- \* 編集モードでの入力と実行を試してみよう
  1. 下画面にある「編集」ボタンを押す
  2. リスト1-3に書いてある通りに下画面のキーボードから入力
  3. 入力が終わったら下画面の「実行」ボタンを押す
  4. 命令1-3を入力して「Enter」キーを押す

リスト1-3：上画面にキャラクタ見本表示

```
CLS
FOR Y=0 TO 15
  FOR X=0 TO 15
    LOCATE X+8, Y+4
    PRINT CHR$(Y*16+X)
  NEXT X
NEXT Y
```

行番号は気にしなくても良いです。

命令1-3：編集モードで入力した命令を実行する

RUN

```
0001 SAMPLE8
0002   ベンツォ ヌイレイ
0003
0004
0005 CLEAR
0006 IF VERSION<&H1050 THEN END
0007 ACLS
0008
0009 --- SORT/RSORTの ヌイレツ
0010 HMAX=5
0011 DIM HENSU(HMAX)
0012
0013 --- ヌイコ ショキカ
0014 MENUHAX=7
0015 DIM MENU$(MENUHAX)
0016 DIM INFO$(MENUHAX)
0017
0018 FOR I=0 TO MENUHAX-1
0019   READ MENU$(I)
0020   READ INFO$(I)
0021 NEXT I
0022
0023 DATA " INSTR"
0024 DATA " キョウシツ ヌサカス カンスウ"
0025
0026 FILE. LOAD. SAVE. CONT. RUN
```



全ての入力が終わったら「実行」ボタンを押して実行モードに切り替えてください。

# 1-4 画面と文字の関係 (ついでに音も)

## \* 文字表示画面のしくみ

- \* 横32文字、縦24文字表示できる
- \* 文字の色は16色から選択可能

命令1-4A : 文字画面を消して右上の方にAを表示

```
CLS:LOCATE 28,5:PRINT"A" ↵
```

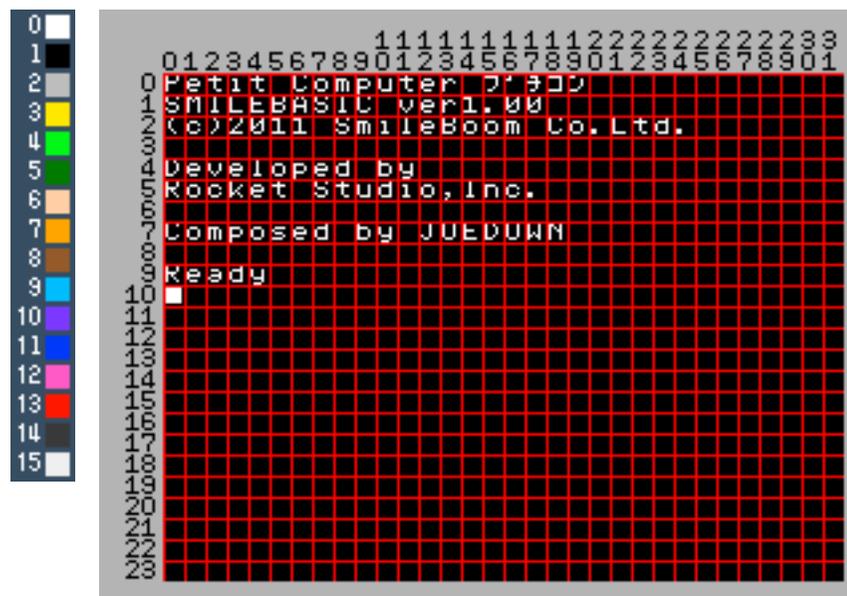
命令1-4B : 文字色を赤にしてNAMEを表示

```
COLOR 13:PRINT"NAME":COLOR 0 ↵
```

## \* 効果音を鳴らす

命令1-4C : 「ぽよ〜ん」という音を鳴らす命令

```
BEEP 8 ↵
```



### 【 ヒント 】

- 音楽を演奏させる・・・ BGMPLAY 番号(0~29)
- 音楽を止める・・・ BGMSTOP
- 効果音の周波数変更・・・ BEEP 番号(0~69), 周波数(±8192)

# お絵かきツールの開発

DSのタッチパネルを使ってお絵かきしてみよう！  
プログラムの意味が分からなくても気にせず入力しましょう。

2

## 【 ヒント 】

危険な命令「NEW」を実行モードで入力すると編集モードに登録してあったプログラムがすべて消えてしまいます。

## 2-1 タッチ操作で線を引く

### \* 線を引くための予備知識

- \* グラフィックは横256ドット、縦192ドット、256色のドットが打てます
- \* DSiのタッチ情報は、TCHST、TCHX、TCHYの変数に入ってきます
- \* タッチされるとTCHSTに1が入ります（通常は0）

### \* 線を引くための動きを考えておく

1. 新しくタッチされた場所を記憶しておく
2. 次のタッチ入力を少し間を空けて待つ（1/60秒）
3. 新しいタッチ位置と記憶された位置の間に線を引く
4. タッチが放されるまで2からを繰り返す
5. 次にタッチされたら1へ戻る

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	<input checked="" type="checkbox"/>	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
1	<input checked="" type="checkbox"/>	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
2	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
3	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
4	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
5	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
6	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
7	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
8	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
9	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
A	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
B	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
C	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
D	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
E	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
F	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■

#### 【 ヒント 】

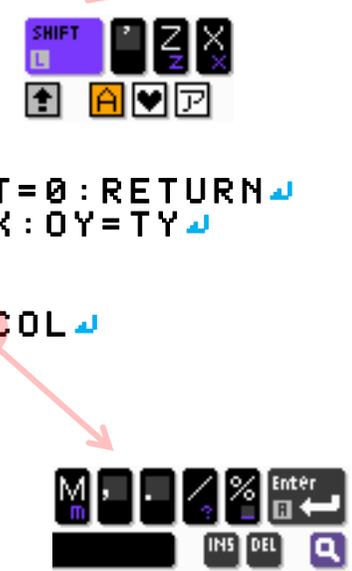
プログラムの中で色番号を管理している変数は、COLです。  
COL=15で白、COL=2で赤、COL=4でピンク。上の色見本が  
色番号の順番に並んでいます。左上が0番、右上が15番、1段下  
がった左2段目が16番、右下が255番となっています。

## リスト2-1：タッチ操作で線を引く

```

CLEAR
'-----
@RESET
GOSUB @DISPINIT
OX=0:OY=0:OT=0:COL=15
'-----
@LOOP
TX=TCHX:TY=TCHY
GOSUB @SETLINE
VSYNC 1
GOTO @LOOP
'-----
@SETLINE
IF TCHST==0 THEN OT=0:RETURN
IF OT==0 THEN OX=TX:OY=TY
'---
@LINE
GLINE OX,OY, TX, TY, COL
OX=TX:OY=TY:OT=1
RETURN
'-----
@DISPINIT
CLS
PNLTYPE"OFF"
GPAGE 1:GCLS 0
SPPAGE 1:SPCLR
RETURN

```



正しく入力されたら実行モードからRUNで実行。  
下画面にタッチで白い線が描けるようになります。  
終わるときは、**SELECTボタン**を押してください。

変数名	用途
TCHX,TCHY	システムが用意しているタッチ座標
TCHST	システムが用意しているタッチ状態 (0=押される前、1=押されている)
TX,TY	TCHX,TCHYの保管用 (ただのコピー)
OX,OY	座標の記憶用
OT	初めて触られたかの管理用 (0=初めて、1=すでに触られている)
COL	色番号

ラベル	用途
@RESET	初期化開始位置
@LOOP	メインループ
@SETLINE	線を引く処理
@LINE	線を引く処理
@DISPINIT	画面関係の初期化処理

## 命令2-1：編集集中のプログラムを保存

```
SAVE "OEKAKI"
```

## 2-2 ボタンを受け取る

### \* ボタンを取得する命令を追加

虫めがねボタンを押す



Q LOOP



探したい文字を入力する LOOP  
十字ボタンの上や下を押すと他の候補へ

#### リスト2-2：ボタン受付処理を挿入

```
:  
@LOOP  
BT=BTRIG( )  
BN=BUTTON( )  
TX=TCHX:TY=TCHY  
GOSUB @SETLINE  
:
```

この赤色部分を  
入力します

### プログラム内の 文字を探す方法

数値	対応するボタン
1	十字ボタンの上
2	十字ボタンの下
4	十字ボタンの左
8	十字ボタンの右
16	Aボタン
32	Bボタン
64	Xボタン
128	Yボタン
256	Lボタン
512	Rボタン



プログラムの途中に  
新しい行を追加するためには  
Enterキーを必要な分押します

変数名	用途
BT	DSiのボタンの状態
OLDBT	1つ前のボタンの状態

※次のページに続きます

## 2-3 Xボタンで画面を消す機能を追加

- \* Xボタンに対応する処理を挿入

Q SETLINE

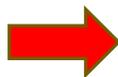


リスト2-3A : Xボタンの押下判断処理挿入

```
:  
@SETLINE  
IF BT==64 THEN @CMDCLS  
IF TCHST==0 THEN OT=0:RETURN  
IF OT==0 THEN OX=TX:OY=TY  
' ---  
:
```

- \* 画面を消去する機能を追加

Q DISPINIT



リスト2-3B : 画面を消す処理追加

```
:  
' ---  
@CMDCLS  
GCLS 0  
RETURN  
' -----  
@DISPINIT  
:
```

※次のページに続きます

ラベル	用途
@CMDCLS	画面消去

## 2-4 十字ボタン操作で色変更機能を追加

### \* 十字ボタンで色を変更する機能を追加

- \* ボタンに対応する色を設定

Q SETLINE



リスト2-4A：十字ボタンの押下判断処理挿入

```
:  
@SETLINE  
IF BT==1 THEN COL=2  
IF BT==2 THEN COL=4  
IF BT==4 THEN COL=11  
IF BT==8 THEN COL=15  
IF BT==64 THEN @CMDCLS  
:
```

- \* 色の情報を上画面に表示

Q LOOP



リスト2-4B：色の番号表示

```
:  
@LOOP  
LOCATE 0,0  
PRINT "COLOR:";COL;" "  
BT=BTRIG()  
BN=BUTTON()  
:
```

命令2-4：編集中のプログラムを保存

```
SAVE "OEKAKI2"
```

# おまけ 1

このページは少し余裕がある人向けのおまけです。  
他の人の作業が終わるまで時間がある人は  
入力して機能を追加してみましょう。

## \* 時間がある人は、他の機能にも挑戦

\* Lボタンが押されてたら塗りつぶし

おまけ1 : ボタン対応と塗りつぶし処理挿入

```
:  
@LINE↵  
IF BN AND 256 THEN @PAINT↵  
GLINE OX, OY, TX, TY, COL↵  
OX=TX:OY=TY:OT=1↵  
RETURN↵  
'---↵  
@PAINT↵  
GPAINT TX, TY, COL↵  
RETURN↵  
:
```

Q LINE

命令おまけ1 : 編集中のプログラムを保存

```
SAVE "OEKAKI3"↵
```



# ゲームの開発

ここからはPチコンでゲームを作ります。  
まずは、どんなゲームにするかルールを決めます。

3

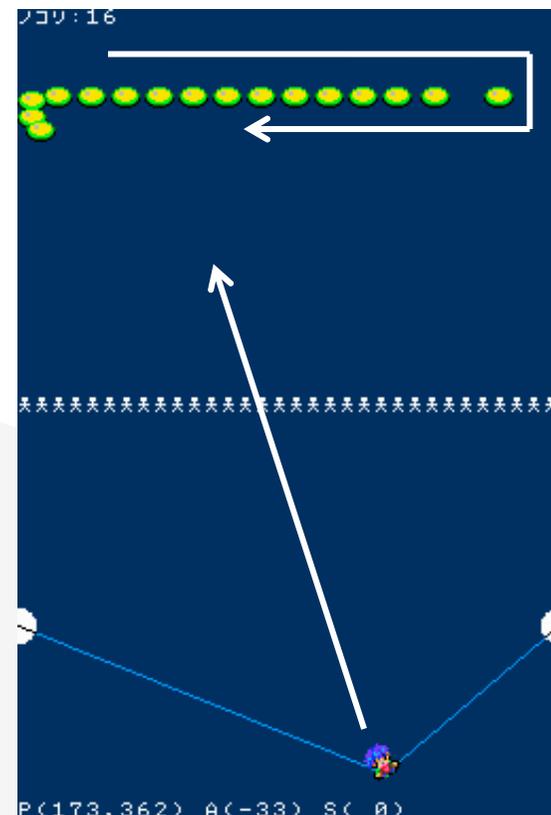
# こんなルールにしてみよう

## \* どんなゲームにしたいのか？

- \* 気軽に遊べて簡単なシューティングゲーム
- \* 下画面のタッチを使った跳ね飛ばす攻撃
- \* 少しずつ降りてくる上画面の敵を体当たりで倒す

## \* どんなプログラムが必要か？

- \* 下画面タッチ操作に合わせた弾発射プログラム
- \* 弾が上画面に飛んでいくプログラム
- \* 上画面で下に降りてくる敵のプログラム
- \* 弾と敵との当たり判定
- \* 全部倒した時と死んだら終了するプログラム



# 3-1 ひとまず基本プログラム

## \* ゲームの基本プログラムです

リスト3-1A : ゲームの基本プログラム

```
'  
' ウィザードプログラムセミナー'  
'  
CLEAR:ACLS:PNLTYPE"OFF"  
' ---  
EMAX=16:ECNT=EMAX  
SY=16:ST=1  
LOCATE 0,23:PRINT "夫"  
' ---  
CY=100:MD=0:SPD=0  
GM=0  
' ---  
@MLOOP  
OLDTS=TS:TS=TCHST  
VSYNC 1  
ON GM GOTO @MLOOP,@GCLEAR
```

リスト3-1B : ゲームの基本プログラム続き

```
' *** ゲームオーバー '  
BEEP 14:BGMPLAY 6  
GPAGE 0:GCLS 2  
LOCATE 12,11  
PRINT "GAMEOVER"  
END  
'  
' *** ゲームクリア '  
@GCLEAR  
BGMPLAY 5  
LOCATE 8,11  
PRINT "MISSION CLEAR!!"  
END  
'  
' *** ステータスヒョウシ '  
@PUTSTAT  
LOCATE 0,0  
PRINT "ノコリ:";ECNT;" "  
RETURN
```

変数名

用途

GM ゲームの状態 (1=クリア、2=死亡)

ECNT 敵の残り数

SY 敵の下に下がるステップ

ST 敵の降りてきた状況

ここで実行してみましょう

## 3-2 発射装置の呼び出しと初期化

- \* 初期化と呼び出しを追加

Q MLOOP

リスト3-2A : 呼び出し挿入

```
:  
GOSUB @SHOTINIT  
GM=0  
' ---  
@MLOOP  
OLDTS=TS:TS=TCHST  
GOSUB @SHOT  
:
```

- \* 発射台初期化プログラム

Q GCLEAR

変数名	用途
MD	発射台の状態管理
LX,LY	発射台のつままれている位置
ANG	発射角度
SPD	発射速度

リスト3-2B : 初期化プログラムの追加

```
※ここは基本プログラムの最後のところ  
:  
PRINT "ソコリ:";ECNT;" "  
RETURN  
:  
' *** ハッシャダ イショキカ  
@SHOTINIT  
GPAGE 0:GCLS &HEF  
GPAGE 1:GCLS &HEF  
GCOLOR 15:Y=CX:R=8  
X=0:GCIRCLE X,Y,R:GPAINT X,Y  
X=255:GCIRCLE X,Y,R  
GPAINT X,Y:GCOLOR 0  
GDRAWMD TRUE  
LX=128:LY=CX  
PX=LX:PY=LY  
GOSUB @PUTLINE  
RETURN  
' ---  
@PUTLINE  
GLINE 0,CX,LX,LY,15  
GLINE 255,CX,LX,LY,15  
RETURN
```

## 3-3 発射装置のプログラム

1回しか  
はじけません

- \* 発射プログラム本体を挿入  
プログラムの一番最後の行に追加します

### リスト3-3A : 発射プログラムの追加

※ここは発射台の初期化プログラムの最後のところ

```
:  
GLINE 255, CY, LX, LY, 15  
RETURN  
'*** ハッシャッ イ  
@SHOT  
ON MD GOTO @1ST, @2ND, @3RD  
'---  
@4TH  
GOSUB @PUTLINE  
IF LY>CY THEN LY=LY-1  
IF LY<=CY THEN LY=CY:MD=0  
GOSUB @PUTLINE  
RETURN  
'---  
@3RD  
GOSUB @PUTLINE  
LY=LY+VY:SVY=VY+1  
IF VY>=0 THEN VY=-1  
LX=LX+FLOOR((128-LX)/2)  
IF LY<CY THEN MD=3:LX=128  
GOSUB @PUTLINE  
RETURN
```

### リスト3-3B : 発射プログラムの続き

```
'---  
@2ND  
GOSUB @PUTLINE  
LX=TCHX:LY=TCHY  
PX=LX:PY=LY+192  
GOSUB @PUTLINE  
'  
ANG=ATAN(CY-LY, 128-LX)  
ANG=DEG(ANG)+90  
'GOSUB @SETPOS  
IF TS!=0 THEN RETURN  
'  
IF ANG<0 THEN ANG=ANG+360  
MD=2:VY=(CY-LY)/4  
X=128-PX:Y=CY-LY  
SPD=SQR(X*X+Y*Y)/4  
IF SPD<2 THEN SPD=2  
IF SPD>16 THEN SPD=16  
BEEP 8  
RETURN  
'---  
@1ST  
IF SPD!=0 THEN RETURN  
PX=128:PY=192+CY:ANG=0  
IF TS==0 THEN RETURN  
MD=1  
RETURN
```

ここで実行してみましょう

## 3-4 プレイヤーの呼び出しと初期化

- \* 初期化と呼び出しを追加

Q MLOOP

リスト3-4A : 呼び出し挿入

```
:  
GOSUB @PLAYERINIT  
GOSUB @SHOTINIT  
GM=0  
' ---  
@MLOOP  
OLDTS=TS:TS=TCHST  
GOSUB @PLAYER  
GOSUB @SHOT  
:
```

- \* プレイヤープログラム追加

プログラムの一番最後の行に追加します

リスト3-4C : 発射装置のコメントアウト解除

```
:  
ANG=DEG(ANG)+90  
GOSUB @SETPOS  
IF TS!=0 THEN RETURN
```

リスト3-4B : 初期化プログラムの追加

※ここは発射台のプログラムの最後のところ

```
:  
MD=1  
RETURN  
' *** プレイヤージョキカ  
@PLAYERINIT  
' --- UI  
SPPAGE 0:SPCLR  
SPSET 0,68,0,0,0,1  
SPHOME 0,7,15  
SPOFS 0,0,-16  
' --- シタ  
SPPAGE 1:SPCLR  
SPSET 0,0,0,0,0,1  
SPHOME 0,7,15  
SPOFS 0,0,-16  
' --- シタニキャラコヒ  
FOR I=0 TO 4*4-1  
SPPAGE 0  
CHRREAD("SPU1",16+I),C$  
SPPAGE 1  
CHRSET "SPS0",I,C$  
NEXT  
RETURN
```

## 3-5 プレイヤーのプログラム

- \* プレイヤープログラム本体を挿入  
プログラムの一番最後の行に追加します

### リスト3-5A : プレイヤープログラムの追加

※ここはプレイヤー初期化プログラムの最後のところ

```
:  
NEXT  
RETURN  
'*** プレイヤー  
@PLAYER  
IF SPD==0 THEN @SETPOS  
'---  
X=SIN(RAD(ANG))*SPD  
Y=COS(RAD(ANG))*SPD  
PX=PX+X:PY=PY-Y  
SPOFS 0,PX,PY  
IF PX<-16 THEN SPD=0  
IF PX>272 THEN SPD=0  
IF PY<0 THEN SPD=0  
IF PY>400 THEN SPD=0  
'---
```

### リスト3-5B : プレイヤープログラムの続き

```
R=SPHIT(0,1)  
IF R==0 THEN @SETPOS  
'---  
BEEP 6,4000  
I=SPHITNO:SPSETV I,2,1  
SPOFS I,PX,-16,8  
SPCOL I,0,0,0,0,0,1  
SPD=0:ECNT=ECNT-1  
IF ECNT==0 THEN GM=1  
'---  
@SETPOS  
'  
SPPAGE 0  
SPOFS 0,PX,PY  
SPANGLE 0,ANG  
'  
SPPAGE 1  
SPOFS 0,PX,PY-192  
SPANGLE 0,ANG  
RETURN
```

ここで実行してみましょう

## 3-6 敵の呼び出しと初期化

- \* 初期化と呼び出しを追加

Q MLOOP

リスト3-6A : 呼び出し挿入

```
:  
GOSUB @PLAYERINIT  
GOSUB @SHOTINIT  
GOSUB @ENEMYINIT  
GM=0  
' ---  
@MLOOP  
OLDTS=TS:TS=TCHST  
GOSUB @PLAYER  
GOSUB @SHOT  
GOSUB @ENEMY  
:
```

- \* 敵プログラム追加

プログラムの一番最後の行に追加します

リスト3-6B : 初期化プログラムの追加

※ここはプレイヤープログラムの最後のところ

```
:  
SPANGLE 0, ANG  
RETURN  
' *** 敵ジョキカ  
@ENEMYINIT  
SPPAGE 0  
X=0:Y=16:V=1  
FOR I=EMAX TO 1 STEP -1  
  SPSET I, 200, 0, 0, 0, 1  
  SPSETV I, 0, V  
  SPSETV I, 1, 0  
  SPSETV I, 2, 0  
  SPOFS I, X, Y: X=X+(16*V)  
  IF X<0 THEN X=0:GOTO @EIREV  
  IF X<256 THEN @EISKIP  
  X=240  
@EIREV  
  Y=Y+SY:V=-V  
@EISKIP  
NEXT  
RETURN
```

## 3-7 敵のプログラム

### \* 敵プログラム本体を挿入

プログラムの一番最後の行に追加します

#### リスト3-7A : 敵プログラムの追加

※ここは敵初期化プログラムの最後のところ

```
:  
@EISKIP  
NEXT  
RETURN  
'*** 予備  
@ENEMY  
SPPAGE 0  
'---  
ET=ET+1  
IF ET%240 THEN @ESTART  
ST=ST+1  
BEEP 55, -2000+(ST*200)  
'---
```

#### リスト3-7B : 敵プログラムの続き

```
@ESTART  
FOR I=1 TO EMAX  
IF SPGETV(I, 2) THEN @ESKIP  
SPREAD(I), X, Y  
V=SPGETV(I, 0)  
W=SPGETV(I, 1)  
IF W=0 THEN @EMOVE  
SPSETV I, 1, W-1  
SPOFS I, X, Y+1  
IF W>1 THEN @ESKIP  
V=-V:SPSETV I, 0, V  
IF Y>=192-24 THEN GM=2  
GOTO @ESKIP  
@EMOVE  
X=X+V*ST:SPOFS I, X, Y  
IF X>240 THEN W=SY:X=240  
IF X<0 THEN W=SY:X=0  
SPSETV I, 1, W  
@ESKIP  
NEXT  
RETURN
```

ここで実行してみましょう

# おまけ 2

このページは少し余裕がある人向けのおまけです。  
他の人の作業が終わるまで時間がある人は  
プログラムを自分なりに改造してみましょう。

## \* 時間がある人は、他のルールにも挑戦

- \* スコアを上画面に表示する
- \* 敵の数を増やしてみる
- \* ハイスコアの追加
- \* 制限時間によるボーナスポイント
- \* 物体の動きの変化（加速？近づいてくる？）
- \* 敵が攻撃してくる



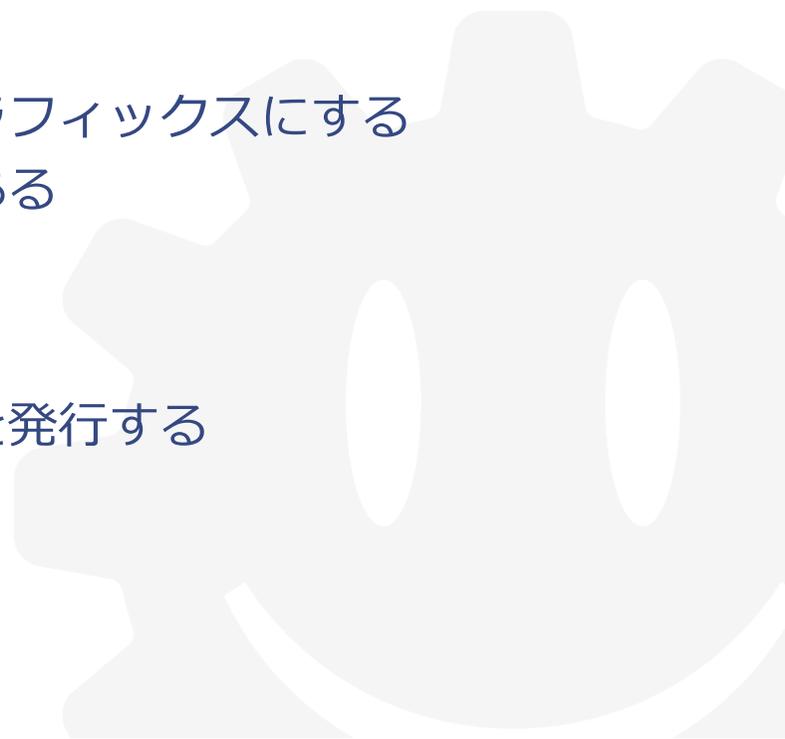
# プログラマーになるには

将来ゲームを作るプログラマーになりたい人へ

4

## 4-1 ゲームの仕事

- \* **プログラマ・プランナ・デザイナーと一緒に作る共同作業**
- \* **プランナ（ゲームデザイナー）の仕事**
  - \* 今までにないゲームのルールを考える
  - \* ハードウェアの特徴を生かした面白い操作方法を考える
- \* **デザイナーの仕事**
  - \* 新しい世界やキャラクターをコンピュータグラフィックスにする
  - \* 平面で表現する場合と立体で表現する場合がある
  - \* ムービーを使った
- \* **プログラマの仕事**
  - \* コンピュータへの命令を理解して適切な命令を発行する
  - \* アイデアや画像データと操作を結びつける



## 4-2 プログラマの仕事

### \* 機械の特徴や性能を生かす仕事

- \* DSにはDSにしかできない性能や特徴があります
- \* WiiにはWiiにしかできない性能や特徴があります
- \* さまざまなハードウェア（機械）の性能や特徴を探ることが一番の仕事
- \* デザイナーやプランナーがゲームを考えるときに機械の性能を伝えます
- \* 誰も見ていない部分で高速化や最適化を進めてニヤリとします

### \* 遊ぶ人の操作を受け取って動きに変える

- \* デザイナーには思いつかないような力技の映像表現を考える
- \* ゲームを作る上で他の職種の夢を現実化する仕事

# おしまい

プログラムは難しい物ではありません。  
BASICは売られているゲーム開発には使われませんが、  
コンピュータへの命令を覚えるには良い言語です。

今回のセミナーをきっかけに  
将来の有望なプログラマが増えることを期待しています。