ゼロからゲームプログラミング

平山　尚

私は何者か

- セガ社員10年。
- 元遺伝子工学。研究から脱落してゲーム屋。

関わったゲーム

電脳戦機バーチャロン
　マーズ

・ボスの攻撃いくつか
・前作の移植部分
・PS2アセンブラコード
・デモシーン再生装置

関わったゲーム

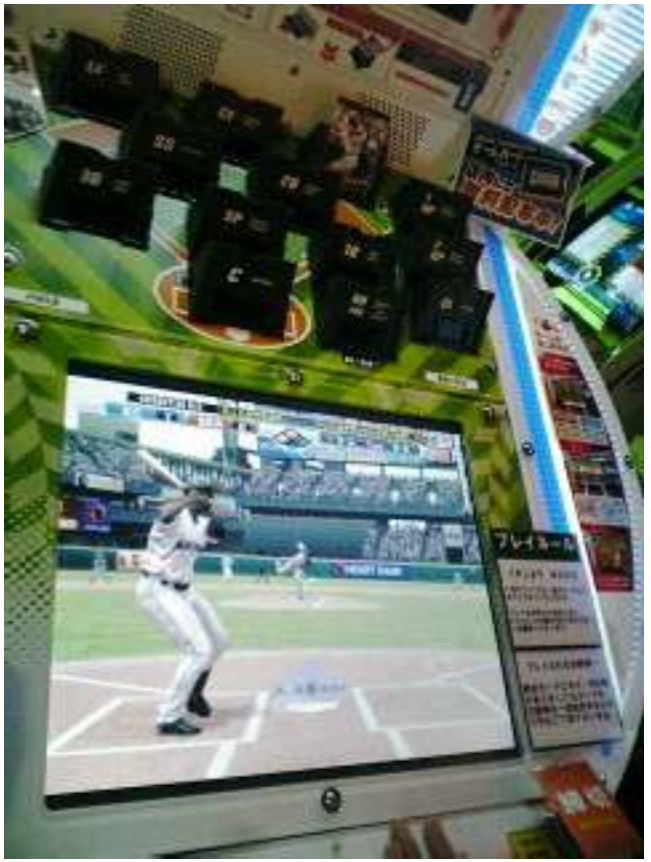
- PowerSmash3/4

- グラフィックス
- ハード叩き
- カメラ制御一部

関わったゲーム

- MLB CardGen

- グラフィックス
- ハードウェア抽象化



本書いてみたり

- この本に書いてあることができれば、だいたいゲーム会社でプログラマをやっても大丈夫だと思います。

この授業の主旨

- プログラミングを**がっつり**学んでもらいます。

何故か？

- 皆さんがプログラマになりたいわけでもないらしい、と言うことは聞いてます。

•だが、敢えてやる。

何故か？(2)

- プログラミングはもはや特殊技能ではない。
  - 楽にプログラミングするための環境
  - コンピュータの使用が多くの職業で前提
    - 例えばゲームの絵かきやゲームデザイナも最近は軽いプログラミングを自分でやる。Unityとか。
- 一方、プログラミングの「根本」をわかってないままになる人が増えている。
  - 文系とか理系とか言う問題じゃない。
  - プログラミングが何なのかを誰も教えてない。

プログラミング教育

- だが、プログラミング教育は未完成というより手付かず。
  - 教わってできるようになった人を私は見たことがない。
  - 独学する気力がある人間のうち運がいい人がたまたまできるようになっている、という以上のものではない。
- なので、この授業でもできるようになることは保障できない。

それでもやる

- 自分がプログラマに向いているかいないかを判定してほしい。
- 向いていない人でも、プログラミングが何なのかだけは理解させる気でいる。
  - プログラマと会話できるようになる。
  - 簡単なプログラミングなら他人に頼まずに済むようになる。

これはプログラミングではない

- 言語の機能を覚えること
- 関数の使い方を覚えること
- 他人の書いたものをいじること

- 結局、自力でやった経験がどれだけあるかが問題になる。
  - 自分で作るな使えと云っているベテランは大抵若い頃自分で作っていた経験がある。信じるな。

今回の授業

- 専用言語でゲームを作る。

早速デモ

- こんな感じのものを作れたらオーケー。
  - テトリス以外に作りたいものがあればそれで。
- 飾り付けをたくさんしたバージョンも用意した。

専用言語？！

- 実用言語は覚えることが多すぎる。
- 実用言語はできることが多すぎる。
- 実用言語だとやってる奴が強い。
- だが専用なら、
  - 覚えることは最小
  - できることも最小
  - 全員同じラインで始まる
    - ぐぐっても無駄よ。

専用言語の特徴

- メモ帳と実行プログラムだけあればいい。
- 簡単に画面に何かを描ける。
- 文法は1時間で覚えられる(人によっては10分)

- 基本的に、何も用意しない。
  - ex.乱数？printf？作れば？

専用言語の特徴(2)

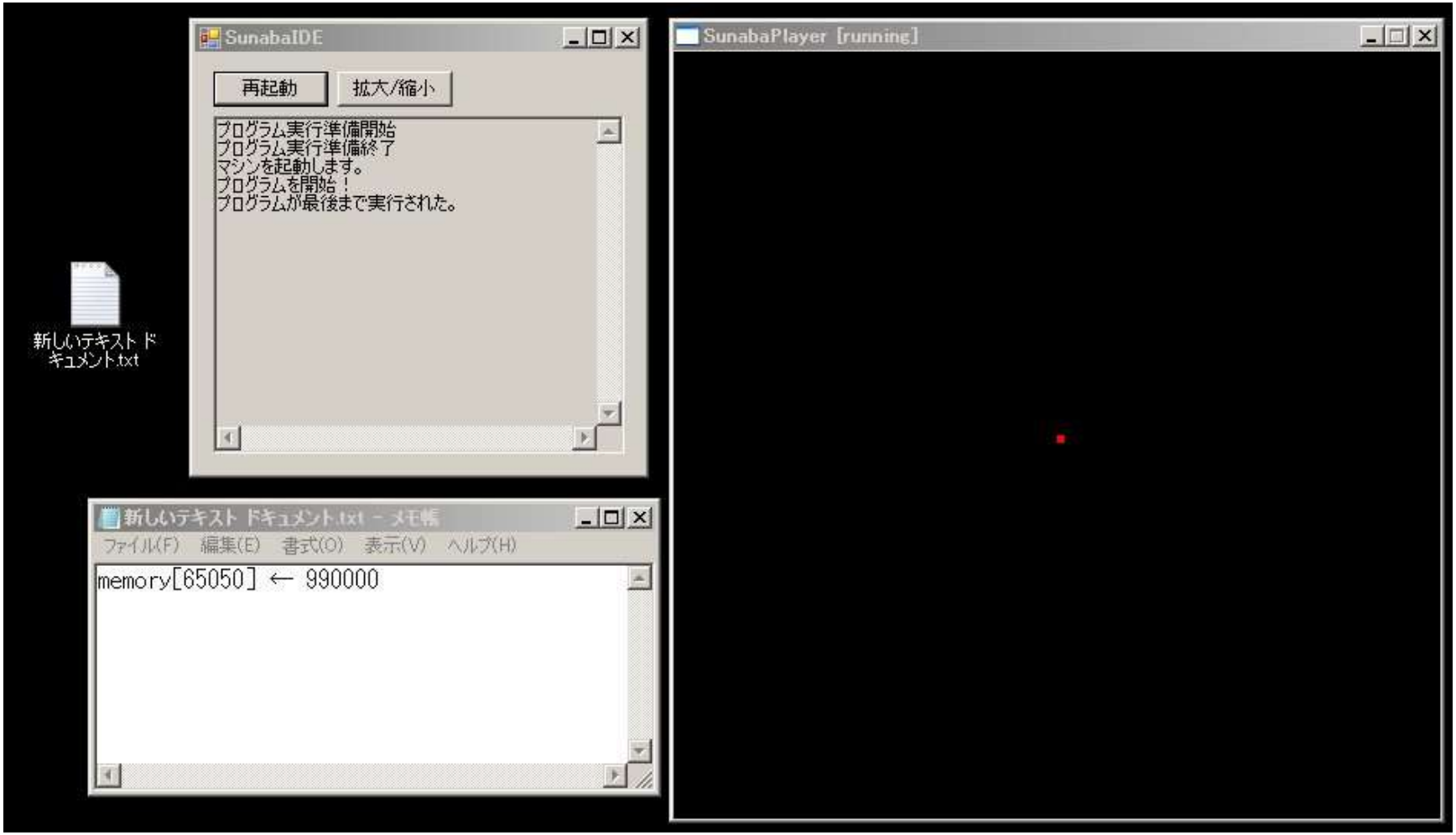
- C系とpythonの折衷
- 日本語を結構使える
- コンピュータの体温を感じてほしい
  - クラス？オブジェクト？それは貴方の心の中にあればいい。コンピュータに見えるのは01です。
- 完成度が低いように見えるが、わざとだ。
  - バグは私のせいなので、直しますが。

とりあえず動かそう

- zipをD:¥で展開
  - bin¥SunabaPlayer.exeがあればオーケー。
- メモ帳を開く
  - 決まったエディタがある人はそれで。
- 以下のように書いて保存

  memory[65050] <- 990000
- SunabaIDE.exeを起動。
- 書いたテキストをドラッグアンドドロップ。

初プログラムおめでとう！



意味は？

- memory[65050] <- 990000
- memory[x]で、「メモリのx番」
- <-は書き込み。右辺の値を左辺のメモリに書き込む。

- つまり、これは「メモリの65050番を990000にする」という意味。

何故書き込むと点が出る？

- メモリの65050番は画面のある点の色を格納している。
- そこの値が変わると画面の色が変わるようにコンピュータとディスプレイが配線されている。

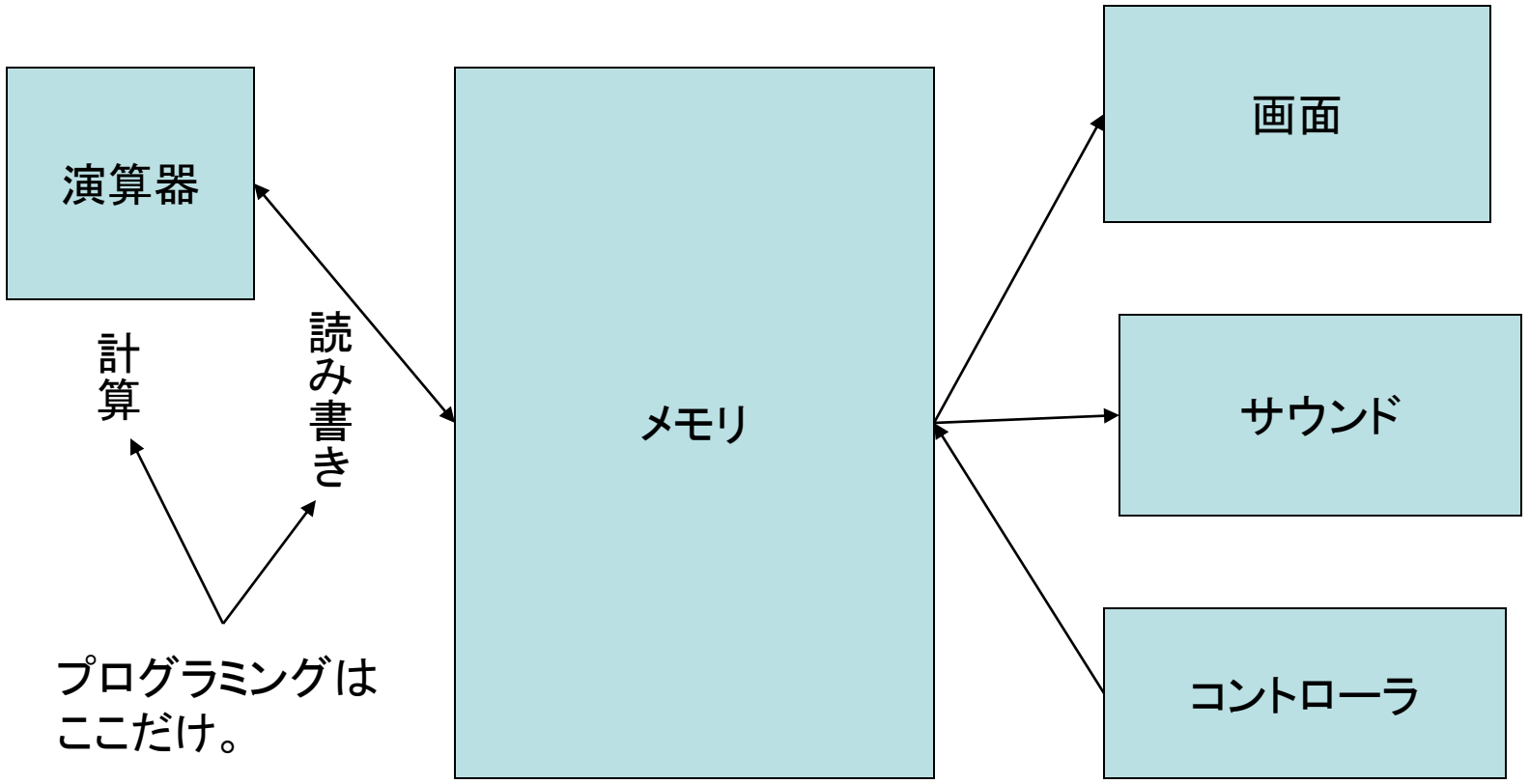
コンピュータは全部これ。

- コンピュータにできることは、「メモリの値を変えること」だけ。
- プリンタもネットワークも画面も、メモリのどこかに配線がつながっていて、メモリの値によって動きが変わる。

ゲーム機も一緒。

- 例えばプレステ2みたいな機械も一緒。
- メモリのx番にポリゴンの色を書き込んで、メモリのy番にポリゴンの座標を書き込んで、メモリのz番に1を入れるとポリゴンを描き始める…のような状態。
- メモリの何番にどういう順番で何を入れるかを制御すること＝プログラミング。

これがコンピュータだ！



これが一番言いたい。

- 正直、これがちゃんと伝わればこの授業の役割は十分果たされたと思ってます。
- 「もうわかったからいいじゃん」と言うかもしれないが、体でわからないと意味がない。

- みんなこう言わないので、私が言う。コンピュータなんてこれだけのものだ。

というわけで、体でわかろう。

- プログラミングに最低限必要な言語要素を紹介する。しばらくこれでいじってほしい。

- zipに入っているSunabaQuickReference.pdfを参照。
  - 印刷して配りますが。

反復

```
while 式
    中身0
    中身1
    …
```

- 中身は行頭に空白を書く。
  - 中身の行頭の空白の数は揃えること。
  - 私は3個をオススメしておく。
- 式を計算した結果0でない限り中身を実行
  - 何回でも繰り返す。

メモリ書き込み(代入)

- 右辺 <- 左辺
  - 「右辺」のメモリの値を、左辺の値にする。
  - memory[0] <- memory[1]
    - 0番の値を1番と同じにする。
- ダイアルがたくさん並んでいるイメージが便利。
  - ダイアルは-10億から+10億くらいまで設定可能
  - 0-9999番は好きに使っていい。

演算子

- +加算,-減算,*乗算,/除算
- 比較(<,>,<=,>=,!=,=)
  - ==じゃなくて=。
  - 成立すれば1、成立しなければ0になる。
- 単純に左から計算。
- 4+5*6→(4+5)*6→54
- 自分で括弧をつけよう！
  - 4+(5*6)→4+30→34

演算子の処理順の例

- 1<2>3!=4=5
- (1<2)>3!=4=5
- (1>3)!=4=5
- (0!=4)=5
- (1=5)
- 0
  - <や=もSunabaにとっては+や*と同じ。

画面につながったメモリ

- 60000-69999：画面の色。
- 縦横100の大きさ。点が1万個ある。
  - 左上から右へ行って、端まで行ったら1段下へ。

y=	x = 0	1		98	99
0	60000	60001	.....	60098	60099
1	60100	60101	.....	60198	60199
	.	.		.	.
99	69900	69901	.....	69998	69999

画面に渡す色

- 0-999999の6桁。
- 「赤赤緑緑青青」
- それぞれ0-99。
  - だいたい%

000000	黒
990000	赤
009900	緑
000099	青
999900	黄(赤+緑)
009999	空色(緑+青)
990099	紫(赤+青)
999999	白(赤+緑+青)

以上を使って

- 画面の上端に赤い線を左端から右端まで引いてください。

線を引く

- 1点だけなら最初にやったように
  - memory[60000] <- 990000
- 左上は60000。右上は60099。
  - この間に全部990000を書けばいい。
- whileを使う。そして、100回やったらwhileから出ていくようにする必要がある。

while

while「まだ100回繰り返していないなら」

memory[60000 +「繰り返した回数」] <- 990000

- 「まだ100回繰り返していないなら」や「繰り返した回数」をプログラムにしてみてね。

答え

```
memory[0] <- 0 //0で初期化して
while memory[0] < 100
    memory[60000 + memory[0]] <- 990000
    memory[0] <- memory[0] + 1 //描いた数+1
```

- 本当に100回描く？
  - 5とかで確認してみよう。
  - 数学的帰納法の応用だよ。
- いい忘れたが、//の後はコメント。

次。

- 左上端に10x10の四角形を描いてください。

四角＝たくさんの線

- 面を短冊に切れば線の集まりになる。
- 線を繰り返し書けばいい。

四角

while「線を10回描いてない」
　「線を描く」

- 日本語で書けばこうだよな？
- 「10回描くまで」はさっき線を描く時にやったはず。

四角

```
memory[0] <- 0
while memory[0] < 10
    「線を描く」
    memory[0] <- memory[0] + 1
```

- あとは「線を描く」をプログラムにすればいいね。それはさっきやった。

四角

```
memory[0] <- 0
while memory[0] < 10
    memory[1] <- 0
    while memory[1] < 10
        memory[60000 + memory[1]] <- 990000
        memory[1] <- memory[1] + 1
    memory[0] <- memory[0] + 1
```

これじゃだめだ！

- 線の書き始めは、「n行目の左端」。番号はいくつ？
  - 一番上は60000,次は60100,次は60200…
  - 100づつ増える！

こうだ

```
memory[0] <- 0
while memory[0] < 10
   memory[1] <- 0
   while memory[1] < 10
      memory[60000 + (memory[0] * 100) + memory[1]] <- 990000
      memory[1] <- memory[1] + 1
   memory[0] <- memory[0] + 1
```

繰り返しは基本。

- 繰り返しは基本。コンピュータは繰り返す道具。
- 繰り返しを制御するには繰り返し回数を保存するメモリを使うのが常道。
  - 他にもいろいろ手はあるが、基本はこれ。

- 次は、入力によって処理を変える。

分岐

```
if 式
    中身0
    中身1
    …
```

- 最大1回しか中身を繰り返さないwhile。
  - whileがあれば原理的には必要ない。
    - どうやればwhileで実現できるか考えて欲しいが
  - しかしあまりに便利なので用意してある

メモリの番号

- 0-9999：好きにどうぞ
- 10000-49999：使っちゃだめ
- 50000-69999：機械につながってます
  - 番号によって、読み専用か書き専用
  - 番号によって意味も動作も違う

機械につながったメモリ

- 50000：マウスのX座標が入ってます
- 50001：マウスのY座標が入ってます
- 50002：マウスの左ボタンが押されてれば1、押されてなければ0
- 50003：マウスの右ボタンが押されてれば1、押されてなければ0
  - その他はクイックリファレンス参照のこと

以上を使った例

```
while memory[50003] = 0 //右ボタン
  if memory[50002] = 1 //左ボタン
    memory[60000 + memory[50000] +
      (memory[50001] * 100)] <- 990000
```

- 行頭の空白(緑の下線)の数に注意。
- どこからどこまでがwhileやifの中身なのかは空白の数で見ている。

簡単なお絵かきソフト。

- 前のページのプログラムを書いて実行してみよう。
- マウス左ボタンを押している間、カーソルがある場所に赤い点が打てます。
- 右ボタンを押すと終わります。

改造！

- 速く動かすと途切れる問題。
- 画面の範囲外に出ると異常終了問題。
- その他機能。
  - 色変えられたら楽しいよね。
  - 右ボタンで終了しなくてもいいです。
- 「お絵かき.txt」という名前でよろしく。提出してもらいます。

知っていると便利な機能

- 変数
- 番号アクセス
- 部品(=関数)
- デバグ用文字出力
- ファイル結合

変数

- メモリのある番号に別名をつけられ、これを変数と呼びます。
- 回数 <- 35
- のようにメモリ書き込み文の左にmemory[]以外の名前を書くと、その名前の変数が作られる。
  - 40000番以降で空いているメモリを探してきて、それに名前をつける機能。

別名じゃないよ！

memory[0] <- 10

a <- memory[0]

memory[0] <- 5

aはいくら？

- <-は「ダイアルの目盛を同じにする」の意味。

変数を使おう

- 四角形を描くプログラムを変数を使うように直そう。

例えばこんな

```
lineCount <- 0
while lineCount < 10
   pointCount <- 0
   while pointCount < 10
      memory[60000 + (lineCount * 100) + pointCount] <- 990000
      pointCount <- pointCount + 1
   lineCount <- lineCount + 1
```

番号アクセス

- 変数名[式]と書くと、
- memory[変数の値 + 式]と解釈する。

```
a <- 100
a[1] <- 5
↓↓↓↓
memory[100 + 1] <- 5
```

- メモリの特定の部分に別名をつけても。
  - 画面 <- 60000
  - 画面[20] <- 999999

これを使うとさっきのは

```
screen <- 60000
lineCount <- 0
while lineCount < 10
   pointCount <- 0
   while pointCount < 10
      screen[(lineCount * 100) + pointCount] <- 990000
      pointCount <- pointCount + 1
   lineCount <- lineCount + 1
```

- みやすくするためにおおいに使って欲しい。

部品（関数）

余りを出す(a, b)
　return a - ((a / b) * b)

答え <- 余りを出す(16, 5)

- 「答え」は1になる
- 何回もやることは一回だけ書けばいい。

デバグ用文字出力

- memory[50023]に文字の番号を入れると、SunabaIDEのウィンドウに文字が出る
- 番号は、Unicodeです。
  - 0-9の数値なら48を足すとそれが出ます。
  - memory[50023] <- a + 48
    - aが0-9なら'0'から'9'が出ます。
- これなしでもデバグはできる
  - 特定条件で青い点を出すとか、まあいろいろ。

こんな感じ

```
memory[50023] <- 48 + 0
memory[50023] <- 48 + 1
memory[50023] <- 48 + 2
memory[50023] <- 48 + 3
memory[50023] <- 48 + 4
memory[50023] <- 48 + 5
memory[50023] <- 48 + 6
memory[50023] <- 48 + 7
memory[50023] <- 48 + 8
memory[50023] <- 48 + 9
memory[50023] <- 31070
memory[50023] <- 10 //newline
```

- 「0123456789神」と表示されます。デバグに使おう。

ファイル結合

include “a.txt”

でa.txtの中身をその場所にコピーする。

- 部品を他のファイルに書いてこれで結合すれば、邪魔でなくなって見やすい。大きいものを作る時は必須。

実はもうゲーム作れます。

- 基本的な道具はそろってます。
- ゲームは無限ループです。
  - 入力、状態更新、出力、でループし続けます。
  - お絵かきソフトにはこの三つが揃っています。
- 好き放題やってください。
  - 途方に暮れた人は相談に乗ります。個別に課題出してもいいし。

お絵かきに飽きたら

- 何でもいいし、ぶっちゃけゲームでなくてもいい。
- 何も浮かばない人はとりあえずテトリスでも。
  - 500行くらい書ければ作れます。
  - doc/book/book.pdf
    - 独習用テキスト。しかし未完成な上に推敲してないのでいろいろおかしい。来週までにはもう少しどうにか。

提出

- 今日書いたものは、とりあえずください。
  - 動いても動かなくても。全員の状態を把握する。
  - 名前でフォルダを作って中に入れてください。
  - お絵かき.txtと、他に何かあれば。
- 面白いものは次回取り上げます。
  - 面白い間違い方なども含めて…
- 動かなくても気にしないように。
- 最終回は全員発表する予定。

終わり

- バグ報告、質問はhirasho0@gmail.com

- 最新のSunabaPlayerは
- http://www.page.sannet.ne.jp/hirasho/
- doc/book/book.pdfは独習用のテキスト。でも途中。

おまけ

- 以下おまけの話。

ゼロから数える

- なんでゼロから数えるの？
- 15歳は15年生きてます。
- 100年1月-紀元前100年1月=199年間。
  - 紀元0年があればこんなことにならなかった！

- 0から数えると、「距離」が簡単に出ます。
- その代わり、最後がn-1になります。
- トレードオフです。多くの言語は0からですね。

変数の使い方

- 変数といっても、計算に使うものばかりを入れるとは限りません。記号と考えることが重要。
- 色 <- 0 //黒
- if 色番号 = 0
- 　色 <- 000099 //0番なら青
- if 色番号 = 1
- 　色 <- 990000 //1番なら赤
- if 色番号 = 2
- 　色 <- 009900 //2番なら緑

プロへの壁

- 倉庫番は200行くらいで書けます。
- テトリスは500行くらいで書けます。
- ----------超えられない壁----------
- PowerSmash3は60万行
- バーチャロンマーズは80万行
- 200万行を超えてるゲームもあるなあ。
- Windows2000は数千万行だとか。

どうすれば大きくできる？

- ここで初めてオブジェクトやらクラスが生きてくる。
  - 私一人で毎回1-2万行は書いてます。少ない方。

- あと、人と話す能力ね…
- ゲームデザイナがプログラミングを理解してないと、大きいプロジェクトでひどいことになります。