Move であそぼう！

～デスクトップを動きまわるゴーストを考える～

話者： buynowforsale

はじめまして！

- 前にいる人：
  buynowforsale
  - ゴースト作ってます
  - 作ったの
    - エイミとチー兄ぃ
    - スペースインベーダー

デスクトップを動きまわるゴーストって？

- The Hand（ ¥![move,~] ）
- ゆっくりさんぽ（ ¥![move,~] ）
- （・∀・ ）（ wmove.dll ）
- とことこシリーズ（ winpos.dll ）

¥![move,~] って どんなスクリプトなんだろう

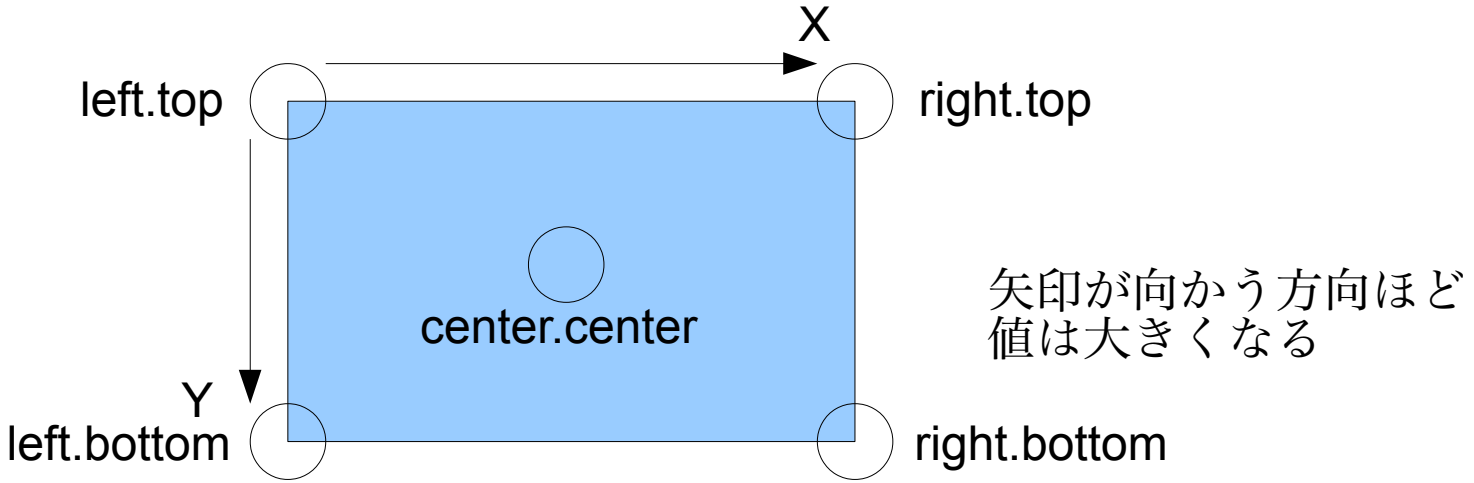
- デスクトップの指定座標へスコープしている キャラクターを移動させる
  - 例１： ¥0¥![move,200,fix]
    - ¥0 側のキャラクターが画面左上を 基準点 (0,0) とし (200, 元の座標) へ移動する
  - 例２： ¥0¥![set,alignmenttodesktop,free]¥! [move,200,200]
    - ¥0 側のキャラクターが画面左上を 基準点 (0,0) とし (200,200) へ移動する

¥![move,~] って どんなスクリプトなんだろう

- 移動完了までの時間を指定することも可能
  - 単位はミリ秒で指定する
  - 例： ¥0¥![set,alignmenttodesktop,free]¥![move,200,400,1000]
    - ¥0 側のキャラクターが画面左上を基準点 (0,0) とし 1 秒 (1000 ミリ秒) かけて (200,400) へ移動する

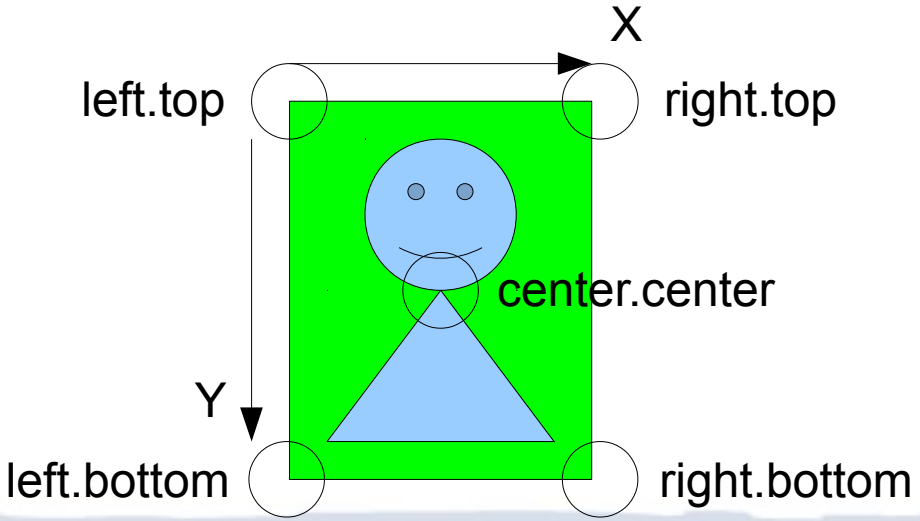
¥![move,~] って どんなスクリプトなんだろう

- 指示する座標の基準点も指定できる
  - 例１ ： ¥0¥![set,alignmenttodesktop,free]¥![move,200,-400,0,screen,left.bottom]
    - ¥0 側のキャラクターが画面左下を
      基準点 (0,0) とし (200,-400) へ移動する



¥![move,~] って どんなスクリプトなんだろう

- 指示する座標の基準点も指定できる
  - 例 1 ： ¥0¥![set,alignmenttodesktop,free]¥![move,50,50,0,1,center.center]
    - ¥0 側のキャラクターが ¥1 のサーフェス画像中心を基準点 (0,0) とし (50,50) へ移動する



¥![move,~] って どんなスクリプトなんだろう

- 指示する座標の基準点も指定できる
  - 基準となるキャラクターは ID 以外にも
    自分自身を指定したり (me) 、
    「 (Sakura 名)/(ID) 」 で他のゴーストも指定できる
  - キャラクターを利用した基準点指定について、
    surface.txt 内の point.basepos 指定を
    用いることができる
    (base.base,screen 指定時は使用不可)
  - 省略時は screen,left.top

¥![move,~] をどう使おうか

- 例：ゴーストの演技として
  - ¥0¥s[3] なんか秋も深まってきたけど、そろそろ色んな「なんとかの秋」を実行に移さなあかんね。 ¥n¥n

    ¥1¥s[10] そんな深刻な顔で考えることか。 ¥n¥n

    ¥0¥s[4] どういうんが手軽かな…… ¥s[5] ん、食欲の秋、とか？ ¥n¥n

    ¥1¥s[12] その辺が妥当だろうな。 ¥n¥n

    ¥0¥s[6]…… もっと手軽なん見つけた。 ¥s[5] もふもふの秋ー。 ¥![move,0,fix,1000,1]¥n¥n

    ¥1¥s[12] なんだそれ…… ¥s[11] おい！近づくな！やめろ！ ¥s[13] モフるな！やめろぉー！ ¥n¥n

¥![move,~] をどう使おうか

- 例：ゴーストの反応として（里々での例）
  - ＊ OnDisplayChange

    ＄ width （Ｒ１）

    ＊ OnMouseEnter

    （ when

    ，（Ｒ３）==0

    ,\0\s[2] やめろっ！触るな！ \![moveφ,（乱数 0 ～（ width ））φ,fixφ,1000]

    ）

¥![moveasync,~] は ¥![move,~] とは何が違うの

- 扱うパラメータに ¥![move,~] との違いは無い
- さくらスクリプト実行時にキャラクターの移動をスクリプトの実行に同期させるかさせないかの違いがある

¥![moveasync,~] は ¥![move,~] とは何が違うの

- ¥0¥s[5]¥![move,200,200,1000] おあよ〜
  - ¥0 にスコープが当たる
  - ¥0 のサーフェスが 5 に変更される
  - 指定の場所に 1 秒かけて移動
  - おあよ〜
- ¥0¥s[5]¥![moveasync,200,200,1000] おあよ〜
  - ¥0 にスコープが当たる
  - ¥0 のサーフェスが 5 に変更される
  - 指定の場所への移動と「おあよ〜」が同時に行われる

¥![moveasync,~] をどう使おうか

- 例
  - ¥0¥s[5] なーみてみてー。買い物帰りにたこ焼き買うてきたんよー。 ¥n¥n

    ¥1¥s[10]……¥n¥n

    ¥0¥s[3]……¥s[7] むっ。 ¥w9¥w9¥n¥n

    ¥![moveasync,50,fix,50,me]

    ¥1¥![moveasync,50,fix,50,me]¥_s ザッ ¥n¥_s¥w[1000]

    ¥0¥![moveasync,-50,fix,50,me]

    ¥1¥![moveasync,-50,fix,50,me]¥_s ザッ ¥n¥n¥_s¥w9¥w9

    ¥0¥s[4]…… あげへんからね。 ¥1¥s[11] むぐぐ。

¥![move,~] と SHIORI Event を組み合わせよう

- 例：スペースインベーダー
  - OnDisplayChange でデスクトップの大きさを取得
  - OnSecondChange で一秒ごとに

    ¥![moveasync,~] で場所を指定する
    - 画面端 / 下に来れば OnDisplayChange で得た情報を用い折り返し処理

¥![move,~] と SHIORI Event を組み合わせよう

- OnSecondChange
  - 一秒ごとに発生する時間イベント
  - ふだんはあまり弄ることがない (色々と面倒なので)
  - スペースインベーダーでは一秒ごとに

    ¥![moveasync,~] を呼び出して

    デスクトップ上をちょこまかさせるために使用

¥![move,~] と SHIORI Event を組み合わせよう

- OnSecondChange
  - * OnBoot

    $ 現在 X 【タブ】 0

    $ 現在 Y 【タブ】 0

    \0\![moveasync,（現在 X ）,（現在 Y ）,0]

    * OnSecondChange

    $ 現在 X 【タブ】（ calc,（現在 X ）+15 ）

    $ 現在 Y 【タブ】（ calc,（現在 Y ）+0 ）

    \0\![moveasync,（現在 X ）,（現在 Y ）,0]

¥![move,~] と SHIORI Event を組み合わせよう

- OnDisplayChange
  - 「 Change 」と付いていますが

    起動時 Notify イベントでもあるので

    起動時のデスクトップの大きさを取得することができます
  - スペースインベーダーでは折り返し時に

    用いるデスクトップ幅と高さの情報を取得する際に使用
  - * OnDisplayChange

    $ width 【タブ】（ R 1 ）

    $ height 【タブ】（ R 2 ）

¥![move,~] と SHIORI Event を組み合わせよう

- 例：スペースインベーダー
  - インベーダーでは以上のことを

    times（里々での繰り返し命令）を用い

    ¥p[0] から ¥p[8] までに

    それぞれ行わせています

動きに変化をつけたい

- 直線移動だけでなく曲線的な移動も出来るとより可愛くなるのでは
  - ふわふわと波を漂うように動いてみたり…
  - 円や渦巻きをくるくると…
    - →YAYA の三角関数を使ってみましょう

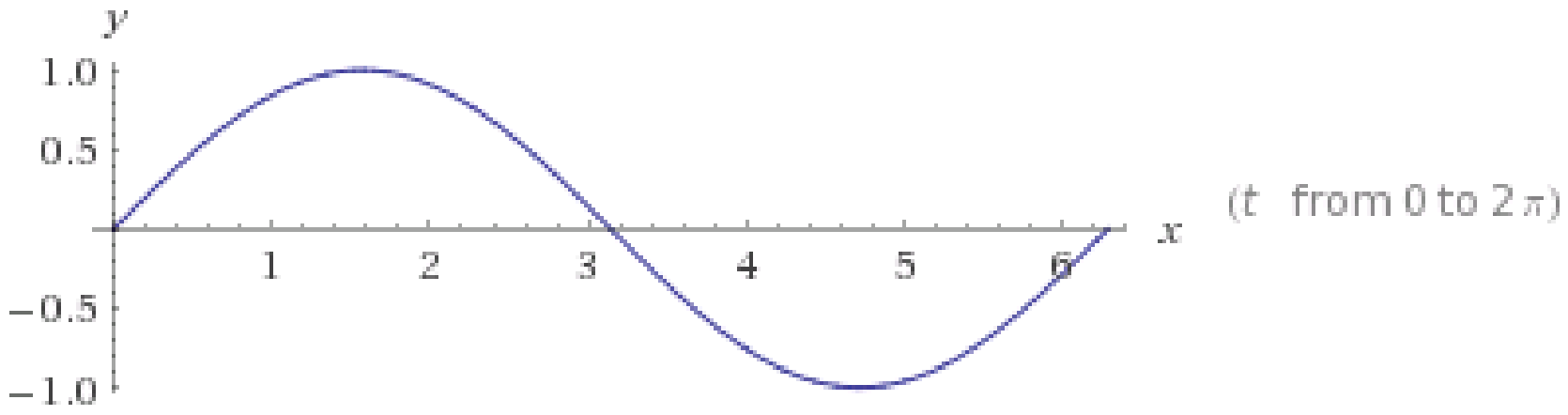
      YAYA には数学用関数として三角関数とその逆関数が用意されています

      SAORI として里々でも使えます

動きに変化をつけたい

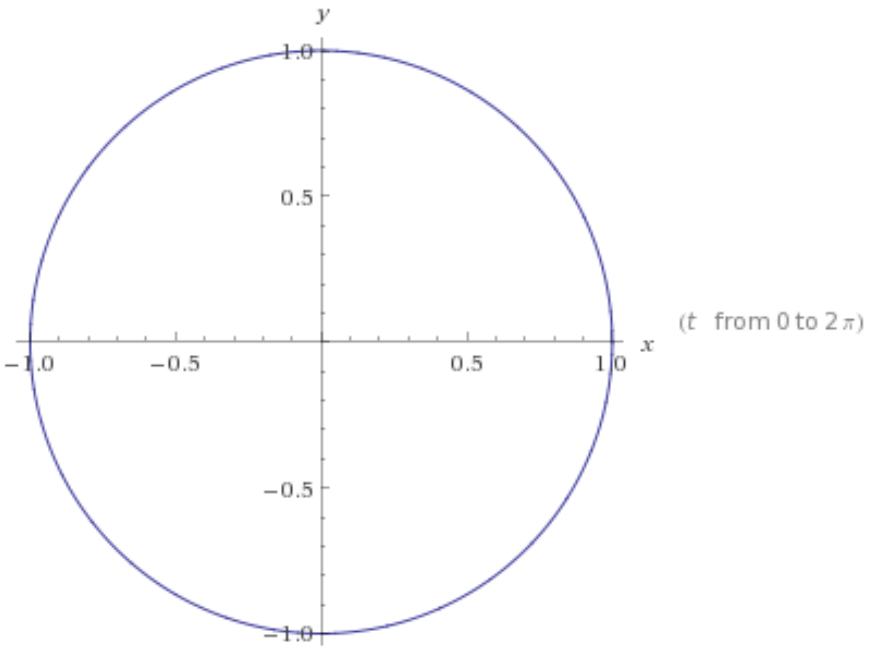
- 例：波
  - x = t

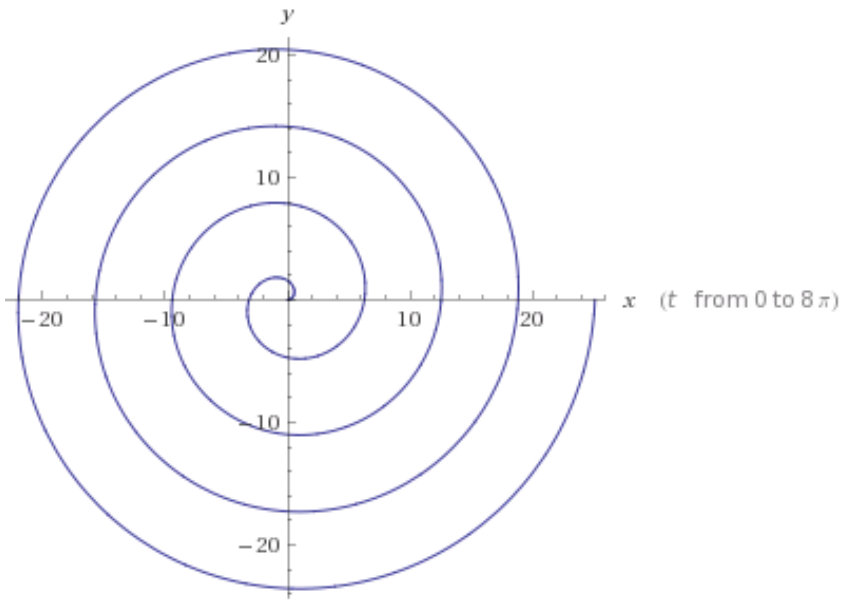
    y = sin t
  - 上記の動きを move に反映させる



動きに変化をつけたい

- 例：円と渦巻き
  - 円： x = cos t, y = sin t
  - 渦巻き： x = t * cos t, y = t * sin t

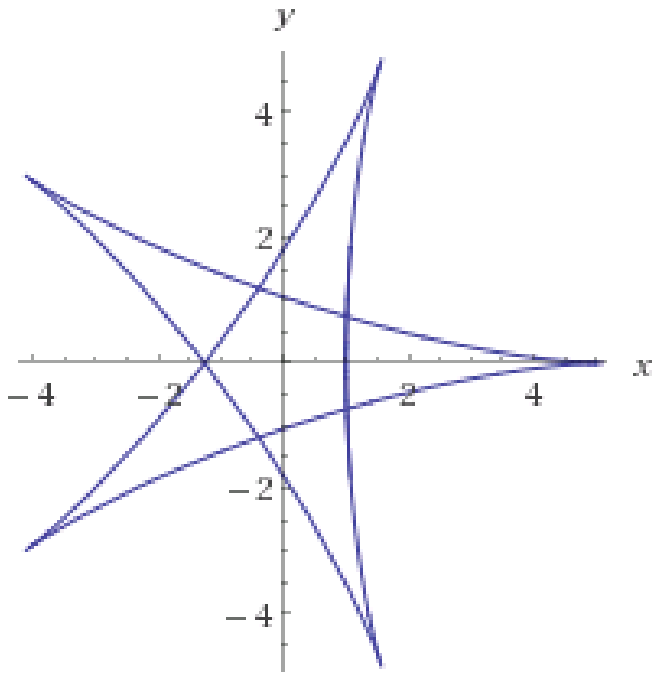




動きに変化をつけたい

- 例：内サイクロイド
  - x= (5-2)cos t + 2 cos(1.5t)

    y= (5-2)sin t - 2 sin(1.5t)



(t from − 6.597 to 6.597)

結語

- ¥![move,~] と ¥![moveasync,~] は単体で使うだけでなく、他のイベントや関数と組み合わせることでより多彩な動きをデスクトップ上で表現する事ができます
- 数学や物理の知識を利用すれば~~よりヘンテコな動き~~でより豊かな動きでデスクトップを動きまわるゴーストが作れるかもしれませんね