

# SCRATCH 入門

スクラッチは小中学生向けに MIT メディアラボが開発した新しいプログラミング環境です。小中学生向けと言っても、最新のプログラミング言語の技術を取り入れた、大変優れたものです。

Scratch は無料で公開され、誰でも以下の専用サイトからダウンロードすることができます。

[www.scratch.mit.edu](http://www.scratch.mit.edu)

## 目次

1. ダウンロードとインストール・・・ 2
2. SCRATCH の画面・・・ 3
3. 習うより慣れよ・・・ 5
4. 命令ブロックの実行規則・・・ 12
5. ステージ作り・・・ 19
6. 変数・・・ 21
7. 配列・・・ 26
8. スプライト間の通信-ブロードキャスト・・・ 32
9. その他の便利な機能・・・ 37

# 1. ダウンロードとインストール

次の手順で、ダウンロードとインストールをします。

Step1: <http://scratch.mit.edu/> へ行って、

Step2: 「Download Scratch」 ボタンを押すと、

Step3: 登録フォームが出てくるので、書き込んでもいいし、書き込まなくても問題なし。

Step4: 登録フォームの下の方の「Scratch download」をクリック

Step5: 次の Widows 版をダウンロード

++2010/3/15 現在は以下のバージョン++

Scratch Installer for Windows

Compatible with Windows 2000, XP, Vista, and 7

[ScratchInstaller1.4.exe](#)

+++++

Step6: 次のインストーラアイコンをWクリックしてインストール開始。

(インストール先を USB メモリに指定する。以下は USB メモリ内)



図1. 1 USB メモリ内にインストールした結果

(インストール時に、デスクトップにアイコンを作るか聞いてくるので、チェックを消して作らない)

Step7: 図 1.1 のフォルダー内の「猫顔マーク」をWクリックすると SCRATCH が起動

(注：自宅でインストールするには、Cドライブにインストールするとよい)

## 2. SCRATCH の画面

図2. 1はSCRATCHのユーザインタフェース画面です。画面は4つの部分に分かれています。

- (1) 舞台 (ステージ) : 役者 (スプライト) が動き回る場所
- (2) 楽屋入口ドア (スプライトの一覧) : 各役者は自分の楽屋を持っていて、右下に楽屋入口のドアが並んでいます。
- (3) 楽屋の中 (スクリプトエリア) : 楽屋のドアをWクリックすると楽屋の中が表示されます。  
楽屋には(a)台本 (スクリプト)、(b)衣装、(c)音、が置いてあり、上部のタブをクリックすると中身が見えます。
- (4) 命令リスト (ブロックパレット) : 台本 (スクリプト) には、役者の動きや言葉が並んでいますが、台本で使える文句 (命令ブロック) には制限があり、どんな文句 (命令ブロック) でも使えるわけではありません。  
使える命令のリストが左の方に並んでいますので、ここから命令ブロックをドラッグしてきて、台本を作ります。

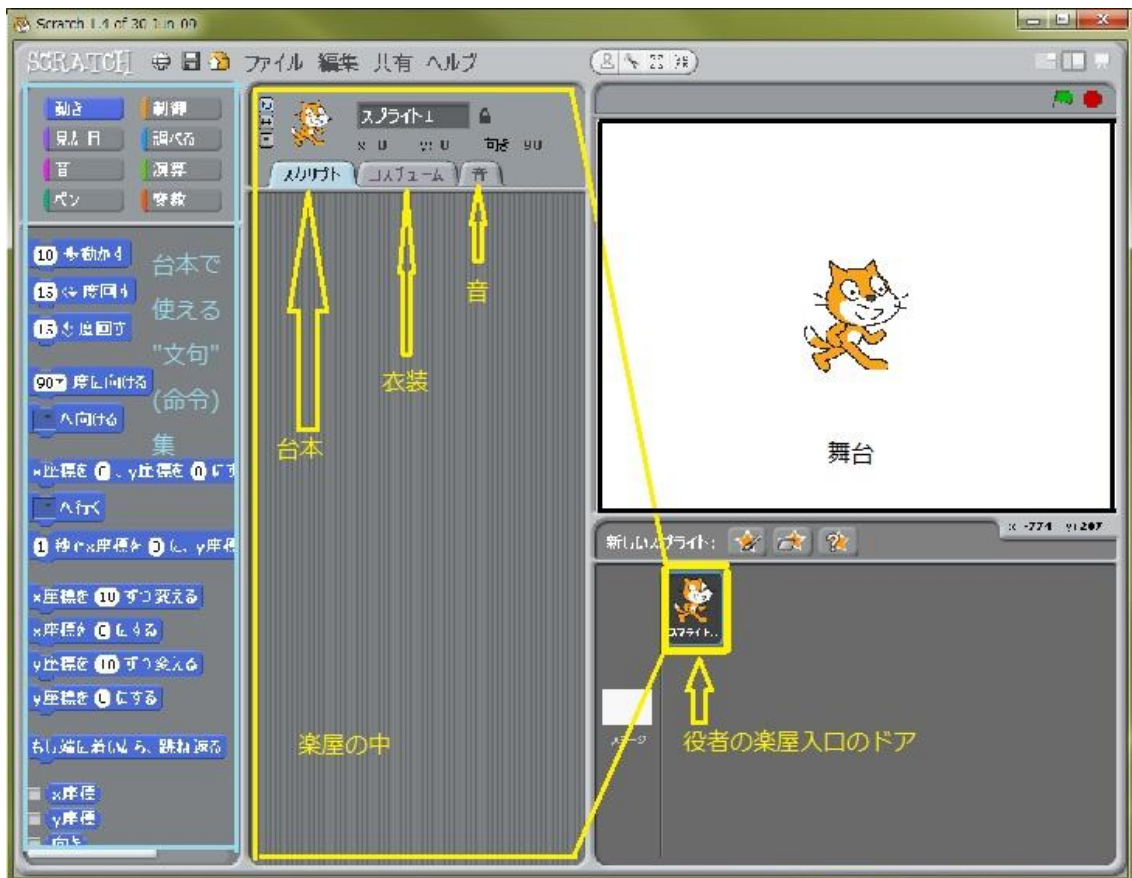


図2. 1 SCRATCH のインタフェース

舞台（ステージ）上の  $x$ 、 $y$  座標は次の通りです。画面中央が原点  $(0,0)$  ですので、注意してください。

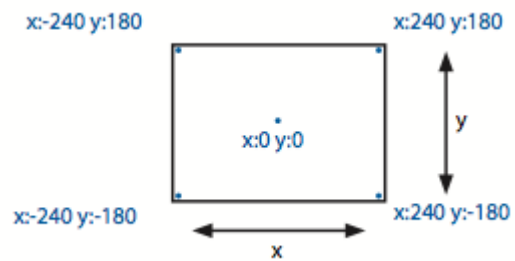


図 2. 2 画面の座標

初期画面には、役者の猫が一匹います。まずは、猫を動かして雰囲気をつかみましょう。

### 3. 習うより慣れよ

8個の練習を紹介します。とりあえず、やってみて”感じ”をつかみましょう。

図3. 1はブロックパレットの上部です。スクリプトで使える命令ブロックは沢山あるので、左上のように「動き」、「制御」、..、など8種類に分類されパレットに格納されています。左上のボタンを押すと、その中身が表示されます。図3. 1は「動き」に分類されている命令ブロックを表示しています。

#### 3. 1 猫を動かそう

次のとおり実行してみてください。

- (1) 「(..)歩動かす」の命令ブロックをスクリプト（台本）エリアのドラッグし、
- (2) 命令ブロックをクリックすると猫が動きます。



図3. 1 猫を歩かせる

#### 3. 2 音を足そう

次のとおり実行してみてください。

- (1) 「(..)のドラムを(..)拍鳴らす」をドラッグし「(..)歩動かす」にくっつけます。
- (2) ブロックのどこかをクリックすると、猫が動き、次にドラムがなります。

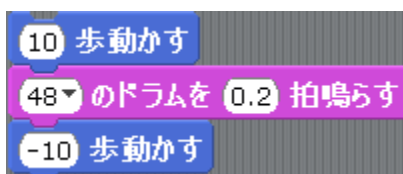


図3. 2 猫を歩かせ、ドラムを鳴らす

- (注1) 動いた直後にドラムが鳴るので、同時に鳴っているように見えますが、動き終わった直後に鳴っています。
- (注2) 「(48▼)のドラムを(...)拍鳴らす」ブロックの▼をクリックすると色々なドラムが選べます。

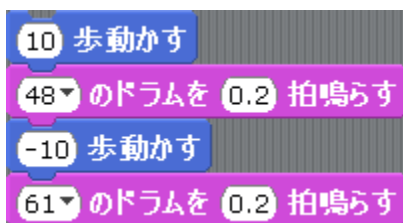
### 3. 3 ダンスをさせよう

(1)もう一つ「(...)歩動かす」ブロックをくっつけ、(...)を-10にします。



くっつけた3つのブロックのどこでもよいかからクリックしてみましょう。前進し、ドラムを叩いて、戻ります。

(2)もう一つ「ドラムを鳴らす」ブロックをくっつけ、クリックしてみましょう。



- (注1) ドラムの種類も”61”に変えてみました。

### 3. 4 何度も何度も（繰り返し）

「制御」に分類されている命令ブロックを使ってみよう。

(1)「ずっと」ブロックを取り出します。





図 3. 3 「繰り返し」を行う

(2)上の4ブロックを固まりにして、「ずっと」ブロックの開いている所へドラッグします。固まりをドラッグするには、一番上のブロックをドラッグします。



(3)どこかのブロックをクリックしましょう。「ずっと」ダンスをします。




「ずっと」の繰り返しを止めるには、舞台右上の  の赤丸  を押します。

### 3. 5 緑の旗（スタートボタン）

「緑の旗がクリックされたとき」ブロックをドラッグして、一番上に取り付けよう。



図3. 4 スタート命令ブロックを使う

このブロックは、舞台右上の   の  と連動していて、緑旗をクリックするとスクリプトが動き出す。

### 3. 6 色を変えよう

今度は、少し変わったことをします。

「見た目」に分類された命令を使ってみます。

「色の効果を(...)ずつ変える」ブロックをドラッグして、これに、

「(スペース▼)キーが押されたとき」ブロックをくっつけます。

スペースキーを押してみましょう。猫の色が変わります。






図 3. 5 スペースキーで猫の色を変える


3. 7 スプライト（役者）を増やそう  
 スプライトを増やすには、




図 3. 6 スプライトを増やす


の中央にある **新しいスプライト:**    の 3 つのボタンを使います。

 は、自分でスプライトを描く時、

 は、フォルダーからスプライトを選ぶ時、

 は、何でもよい時、

に使用します。

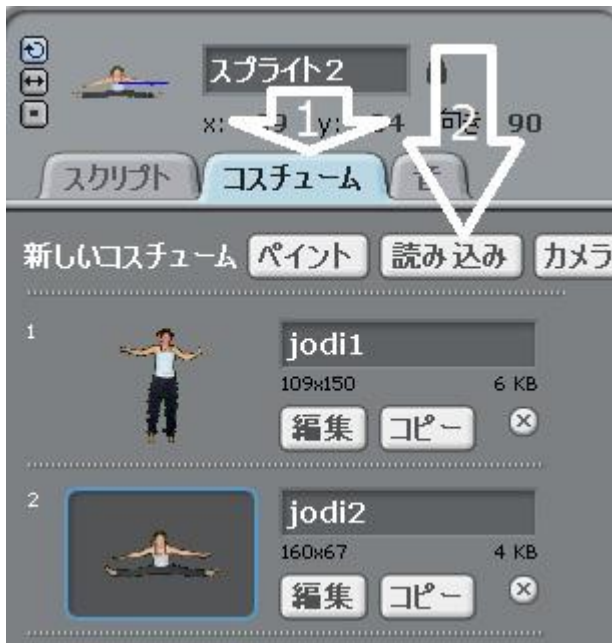
上の図は、 で、People フォルダーを選び、Jodi1 というスプライトを選んだ結果です。猫の隣に Jodi の楽屋入口ができました。楽屋入口がブルーの枠で囲まれている時、左の楽屋内は Jodi の楽屋になっていることに注意してください。



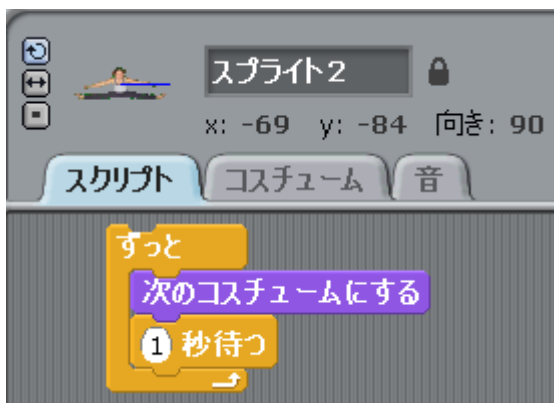
### 3. 8 アニメーション

スプライトのコスチュームを変えることで、アニメーションができます。

Jodi の楽屋の「コスチューム」タブを押し、「読み込み」ボタンを押し、Jodi2 というコスチュームを追加しました。



Jodi のスクリプトを下のようにし、クリックしてみましょう。



Jodi が飛び跳ねます。

感じが掴めたことと思います。

## 4. 命令ブロックの実行規則

お芝居の台本は左端から1列ずつ読まれていきます。スクリプト（以下プログラムと書くこともある）の実行プロセスは図4. 1に示す「連結」が基本で、並んだ命令ブロック（以下命令と略）が上から順に処理される。ただし、「繰り返し」と「条件分岐」という処理順の制御も使えます。

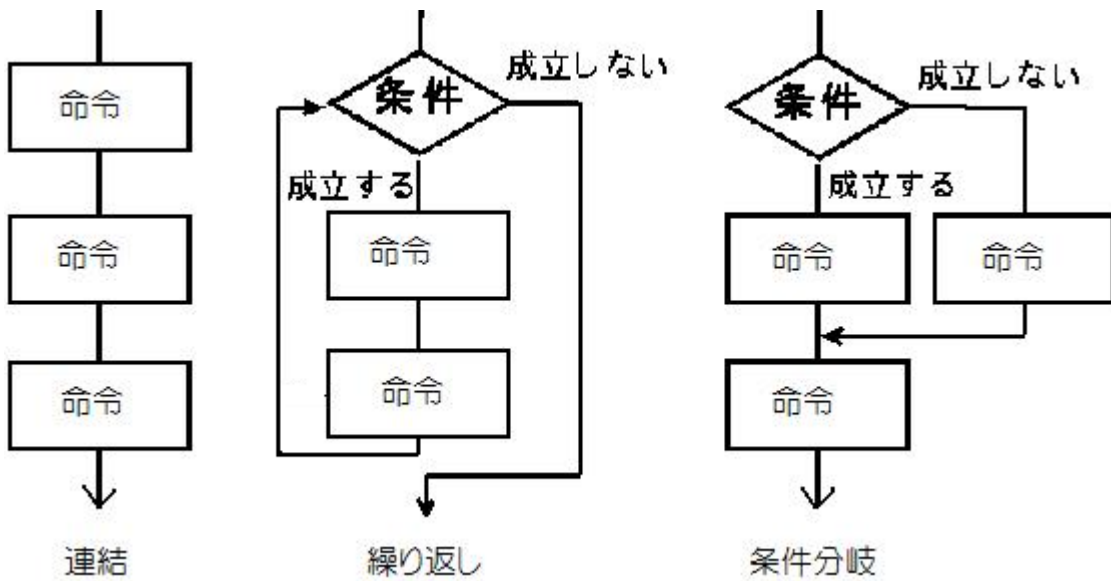


図4. 1 スクリプトの3つの処理形態

別の言い方をすれば、プログラムの基本形態は、この3種類しかありません。全てのプログラムは、この組合せでできています。

### 4. 1 連結

まず、最も基本的で簡単な「連結」の例を示します。くっつけた「命令ブロック」は、上から順に実行されます。



図4. 2 アクション命令の連結

図4. 2のプログラムは、先ずペンを下し、右に100歩動いて、ペンの色を変え、向きを変えて、また100歩移動、します。

#### 4. 2 繰り返しと条件分岐

「繰り返し」と「条件分岐」を実現するには「制御命令」という特殊な命令を使います。

命令は大きく分けると次の3つに分かれます。

- (1) アクション命令：スプライトの動作を指示する命令
- (2) 制御命令：命令の実行順を制御する命令
- (3) 演算命令：計算に使う命令

これらの命令は、図4. 3の8つのボタンに分類されて入っています。

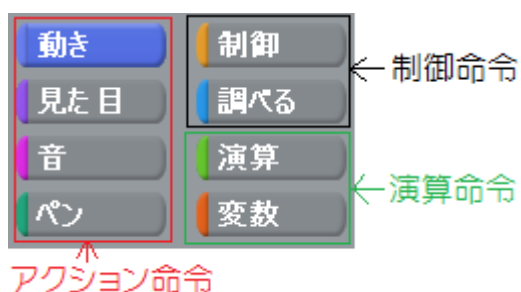


図4. 3 命令の分類

図4. 2で使われた命令はアクション命令と呼ばれ、キャラクターの動きなどを指示します。

スクリプトの命令は上から順に実行されますが、制御命令は、その規則を変更します。制御命令は上図の「制御」ボタンを押すと表示され、大きく分けると次の2つに分かれます。

##### (1) 繰り返し：

繰り返しには、ずっと繰り返す（無限ループ）と、ある条件が満たされている間だけ繰り返す（条件ループ）があります。

SCRIPTでは繰り返す部分を □ の中に書きます。

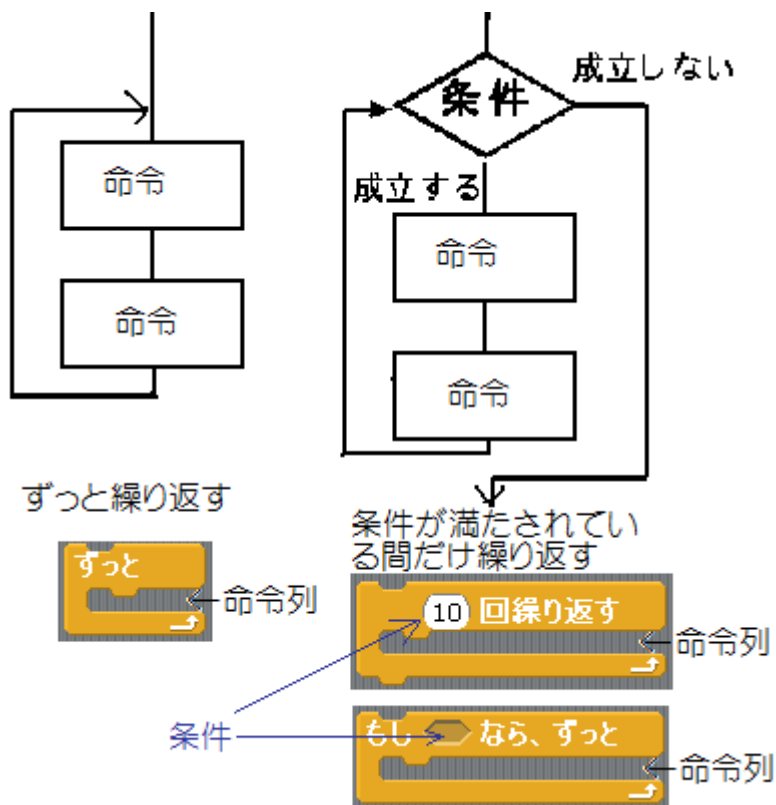


図 4. 4 無限ループと条件ループ

3. 4節では無限ループを使っていました。次に条件ループの例を示します。

猫はx座標が150になるまで右に歩きます。

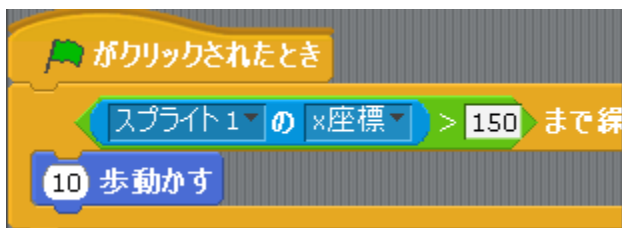


図 4. 5 条件ループの例

条件ループの条件は六角形をしていて、主に「調べる」や「演算」にある六角形のものを使います。

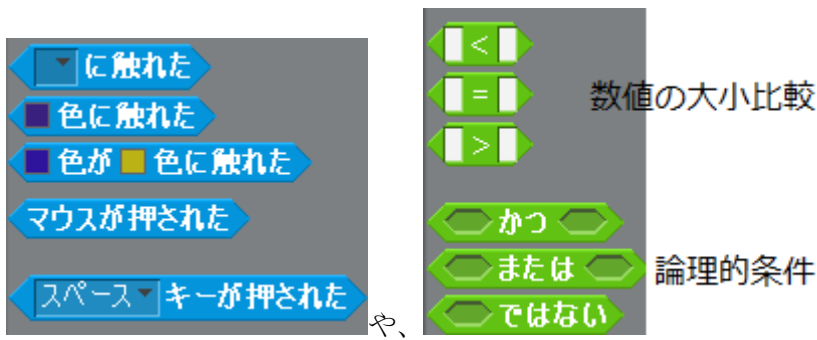


図4. 6 条件に使うもの

(2) 条件分岐：

条件を満たすか否かで、処理の流れを変えます。

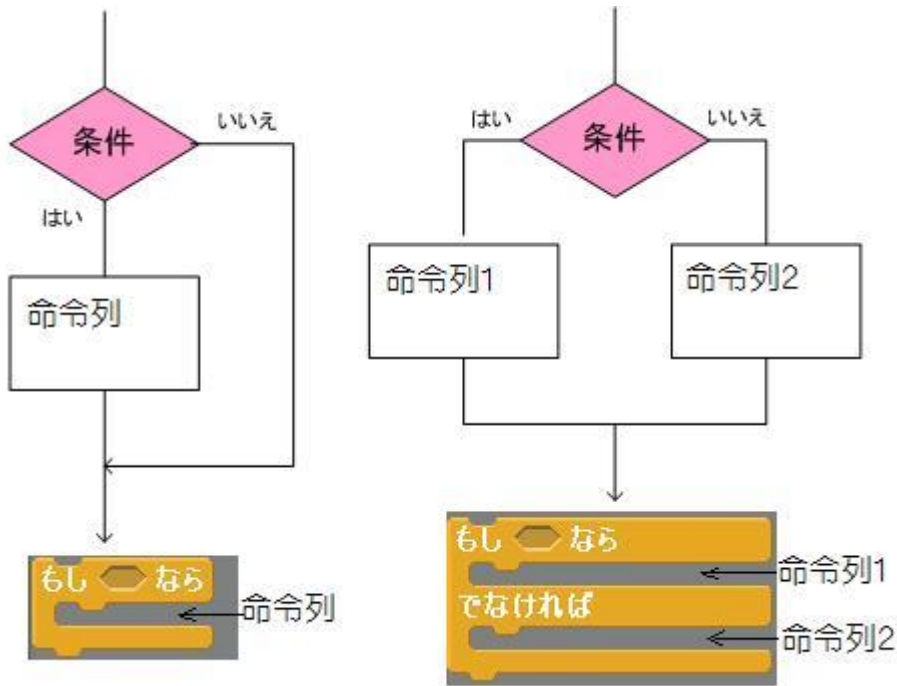


図4. 7 条件分岐の2つのタイプ

「条件分岐」の時の条件も、**調べる** や **演算** で調べます。

2つのプログラム例を示します

(例1) 猫が舞台の上を右に行ったり左に行ったりします。

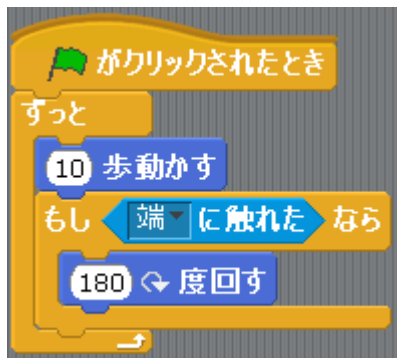


図4. 8 猫の往復

(例2) キーボードの矢印キーで、左右に動かすこともできます。

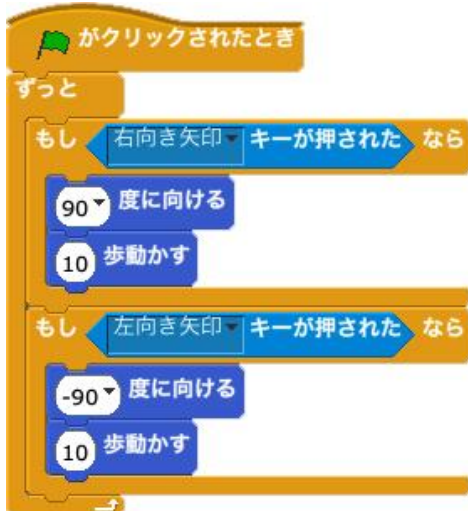


図4. 9 猫をキーボードで動かす

#### 4. 3 2つのスプライトの当たり判定プログラム

3つの当たり判定プログラムを示します。

(例1) 猫は緑の棒に当たると戻ってきます。



図4. 10 猫は緑の棒（スプライト）に当たると戻ってきます。

「もし<□.....>なら」の□の緑色は、□にマウスポインタを持って行って左クリックすると、スポイトマークが出てくるので、それを緑線に持って行ってクリックすると希望の色になります。



(例 2) ライントレース

次のようにすると、灰色の線の上を走る車も作れます。



(例 3) 猫とボールのSpriteがあります。猫は右に歩き、ボールにぶつくと戻ってきます。一方、ボールは猫がぶつかってくると上に飛ばされます。



図 4. 6 猫のスクリプト (プログラム)

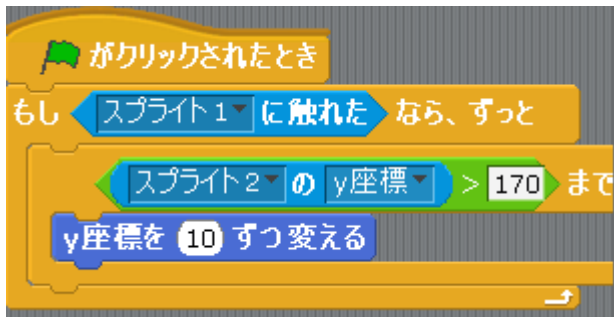


図4. 7 ボールのプログラム

最後にボールを消したければ次のようにします。

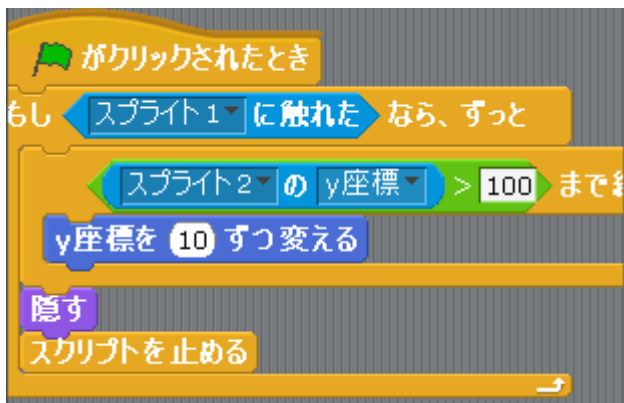


図4. 8 ボールを消すプログラム

また、消えてしまったボールを表示するには、ボールのスクリプトへ行って、「見た目」で、「表示する」をクリック。

## 5. ステージ作り

ステージの背景画を取り換える方法を示します。

今までは、白い背景の上で猫が歩いていました。SCRATCH では、ステージに様々な「背景画」と「音」を用意でき、それらをスクリプトで切替えることができます。

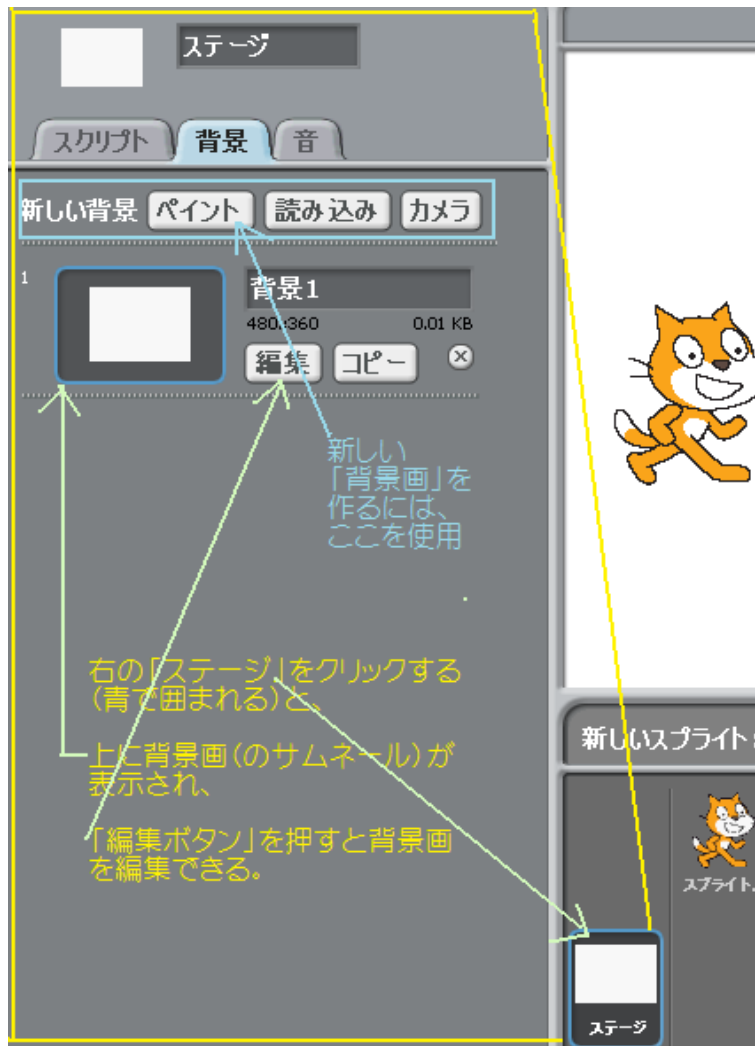


図5. 1 背景画の変更

図5. 1の右下のステージサムネイルをクリックするとステージの「背景画」と背景画を切り替える「台本」が格納された部屋(図5. 1の黄色で囲まれた部分)が表示されます。最初は白地背景なので、白地サムネイルだけが部屋の中にあります。

**Case1:**この白地を他の絵に描き変えたい時は「編集」ボタンを押します。すると、描画環境が表示されるので、ここで描画できます。

**Case2:**新しい背景画を作りたい時は、図5. 1のブルーで囲まれた部分を使います。「ペイント」ボタンを押すと、描画画面が現れ、描画できます。

「読み込み」ボタンを押すと、フォルダーが表示され、色々な背景を選べます。

図5. 1のスク립トでは、背景画を表示するタイミングを指定します。例えば、図5. 2のような2つの背景画を用意しておき、図5. 3のようなスク립トを書いておくと、スタート時にベッドルームの背景が現れ、スペースキーを押すと白い背景に変わります。



図5. 2 2つの背景画



図5. 3 ステージのスク립ト

## 6. 変数

スクリプト（プログラム）ではしばしばデータを扱うが必要になってきます。データを扱うとは具体的には「データの保存」、「データの読出し」、「データの修正」することで、SCRATCH ではこれに「変数」というものを使う。

「変数」は、数値を1つだけ書ける黒板、と考えてください。

### 6. 1 変数を作る

変数は次の順序で作成します。

Step1: 「変数」をクリックすると、「新しい変数を作る」ボタンが現れるので

Step2: クリックすると、右の「変数名設定」ボックスが現れる。

Step3: 変数名を書きこんで「OK」ボタンを押す。

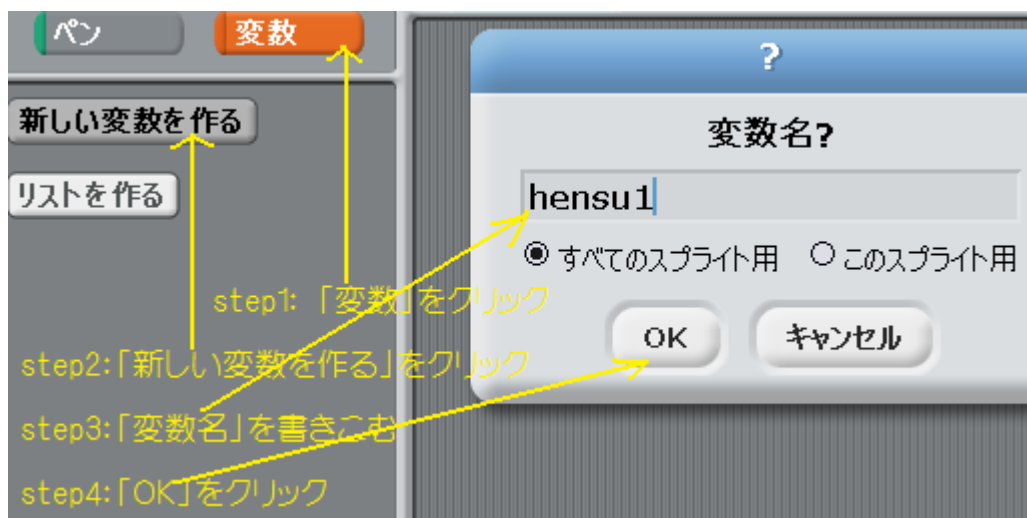


図6. 1 変数の作成

(注) 変数名の下に(1)すべてのスプライト用、と(2)このスプライト、というボタンがある。  
(2)の「このスプライト」とは、スプライトに付属した変数で、  
(1)の「すべてのスプライト用」とは、他のスプライトも共通に使える変数、  
を意味する。

舞台上、変数の値を示す次の図が現れます。



変数のイメージを掴むための例を示します。

(例1) 変数に数値を書き込む (正式には、代入すると言います)

変数に数値を書き込みます。この例では、hensu1 という名の変数に、10 を書き込みます。



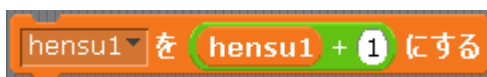
(例2) 変数に足しこむ



変数に数値を加えます。この例では hensu1 という変数に 1 を加えます。この命令をクリックすると、右の hensu1 が 11 に変わります。

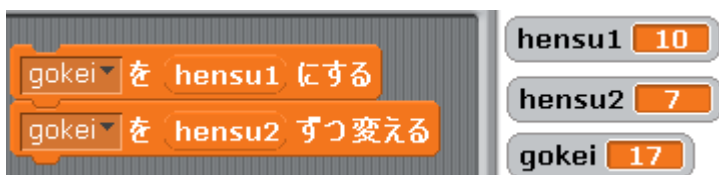
(例3) 変数に定数を加える

例2は、次のようにも書けます。

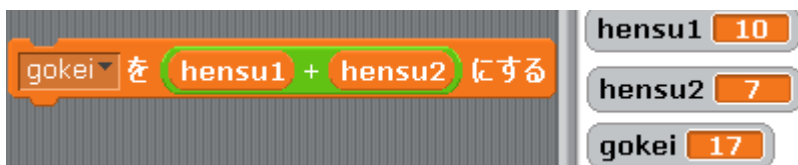


(例4) 変数同士の加算

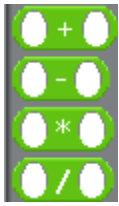
hensu1(10)の値と hensu2(7)の値を足して、結果を gokei という変数に書くには次のようにします。



(例5) 例3は次のようにも書けます。



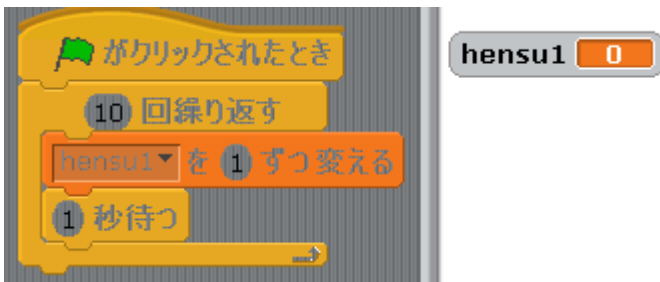
ここで使う「+」は、**演算**の、



を使います。

(例6) 変数の変化の様子を表示

hensu1 という名の変数の値を、1秒ごとに 0,1,2,...,10 と変化させます。



(蛇足) この中の「(hensu1)を(1)ずつ変える」は、“hensu1 に 1 を加える”、ことを意味します。

先ほどの黒板のイメージでいうと、hensu1 という名の黒板に書かれている数値を覚えておいて、それに 1 を加えた数値を黒板に上書きしたことになります。(上書きですので、元の数値は残っていません)

(例7) 二乗を求める

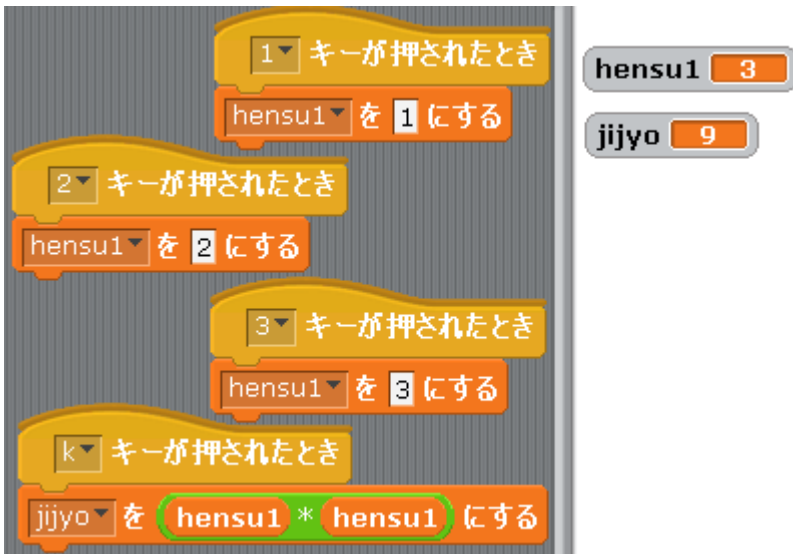
(重要なので再度繰り返します)

「xxx を yyy ずつ変える」

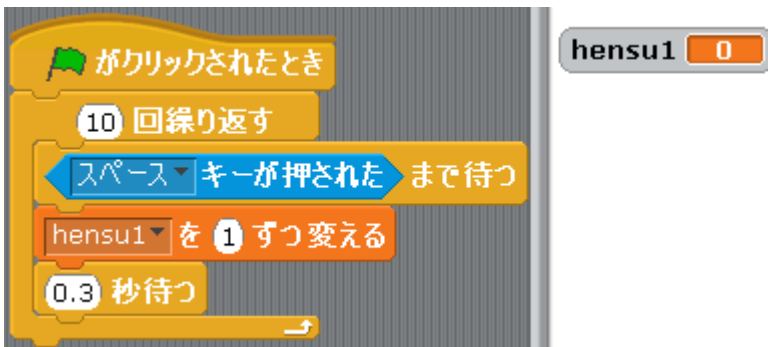


は、xxx という変数に yyy を加えることを意味します。

使える数値は 1, 2, 3 の 3 つだけです。数値キーを押すとその数値が hensu1 に書きこまれ、k というキーを押すと jijyo という変数に hensu1 の二乗が現れます。



(例8) スペースキーを押すごとに hensu1 という変数が1ずつ増える。



(注) 「(0.3)秒待つ」を入れない場合、のろのろスペースキーを押すと次の繰り返しに入っ  
てしまい、1回のスペースキープッシュで hensu1 が2つ以上増えることがある。

(例9) (1～100まで足すプログラム)



```
total=0;
for( i=1; i<101;i++){
  total += i;
```



}

(例 1 0) 猫に 3 段跳びをさせ、全跳ね距離(total)と平均跳ね距離 (average)を求める。

ここでは、**1 から 10 までの乱数** というものを使っています。乱数とはデタラメの数値を言います (コンピュータが作ってくれる)。



The image shows a Scratch script starting with a 'when clicked' event. It initializes 'total' to 0 and 'i' to 1. A loop repeats 3 times. Inside the loop, 'step' is set to a random number between 1 and 10, the cat moves that distance, a 1-second wait occurs, 'total' is increased by 'step', and 'i' is incremented by 1. After the loop, 'average' is calculated as 'total / 3'.

i	4
step	6
total	10
average	3.3

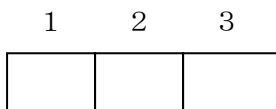


```
total=0;
average=0;
for(i=1; i<4; i++){
    step= random(10);
    moveCat;
    wait(1);
    total += step;
}
average = total / 3;
```

## 7. 配列

配列とは、変数を並べて、その並びに名前を付けたもので、配列の各要素は配列名と添字(インデックス)で指名されます。

例えば、次の図は3つの変数からなる配列で、この配列名を「a」とすると、配列の各要素は、左から、「配列 a の 1 番目」、「配列 a の 2 番目」、「配列 a の 3 番目」、と呼ばれます。



(例えて言えば、配列はアパートで、各部屋が 1 号室、2 号室、3 号室、と呼ばれることに相当します)

配列は「繰り返し」と相性がよく、プログラムの中で頻繁に使われる道具です。

### 7. 1 配列の生成

SCRATCH で配列を作るには、次のように「変数」で「リストを作る」を指定し、変数名を書き込みます (この図で変数名は a)。

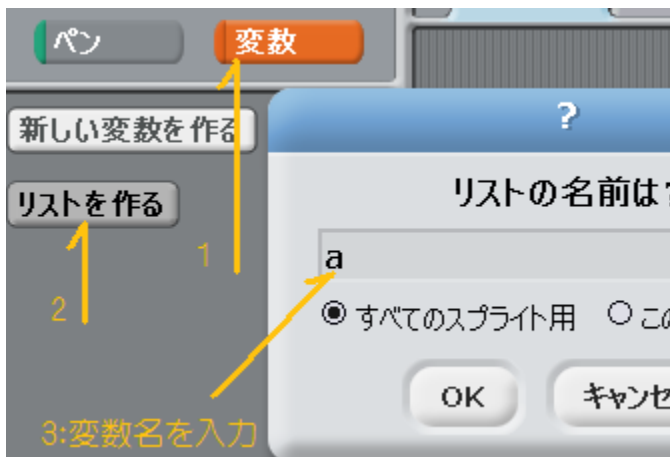


図 7. 1 配列の生成

この段階ではまだ配列は空です。そこで、次のようにすると要素数 3 (各要素の値は 0) の配列が出来上がります。



## 7. 2 インデックスの準備

配列を使うには、配列中の場所を示す添字（インデックス）変数が必要です。配列とインデックス変数はセットですので、配列作成時に一緒に作っておきましょう。次の図は、i という名のインデックス変数を作ったところです。

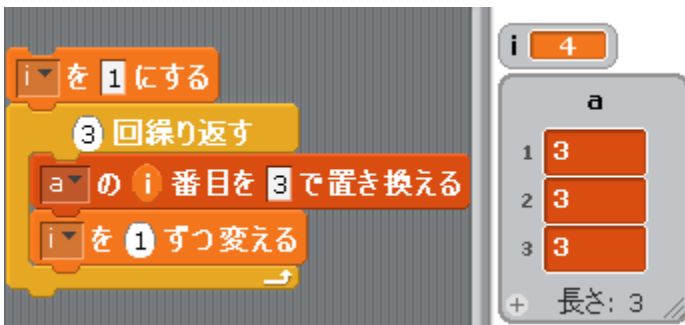


図 7. 2 配列とインデックス

## 7. 3 配列の操作

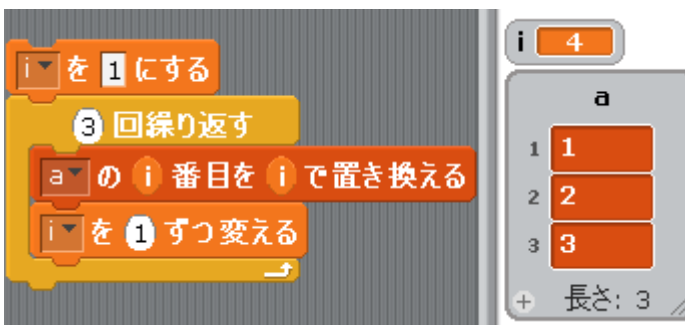
配列操作の例を示します。

(例1) 配列の要素を全て「3」にする。



(右は実行後)

(例2) 配列要素の値を「1, 2, 3」にする。



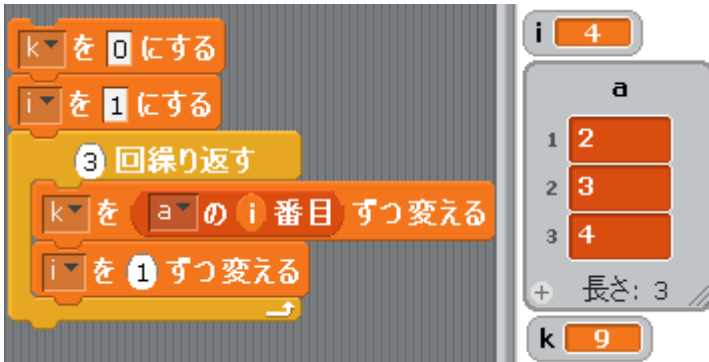
(例3) 配列要素の値を「2, 3, 4」にする。



(例4) もう一つ変数(k)を作り、そこに配列要素の総和を入れる。



または、



(例5) `max` という変数に、最大値を求める。



ここでは、配列の 1 番目の要素を無駄にチェックしていることになります。処理時間を節約するには、次のようにします。

(注：繰り返しが一回少なくなります(`i` は 1 に初期設定しています))



(例6) 最小値を求める。

`min` という変数に最小値が得られます。



#### 7. 4 Java への移行準備


前節の例では、(理解し易いかと考へ) 繰り返し部に「[n]回繰り返す」という命令を使ってきました。



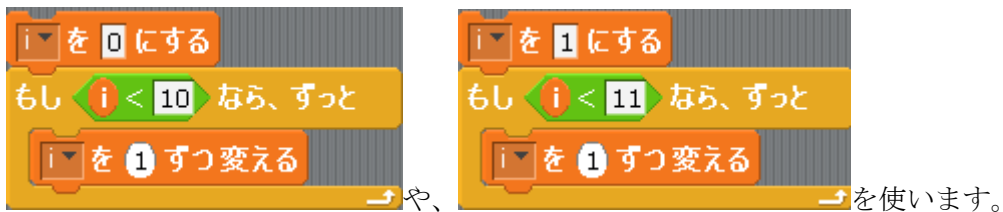
例えば、です。

しかし、Java などのプログラミング言語では、この命令の代わりに、

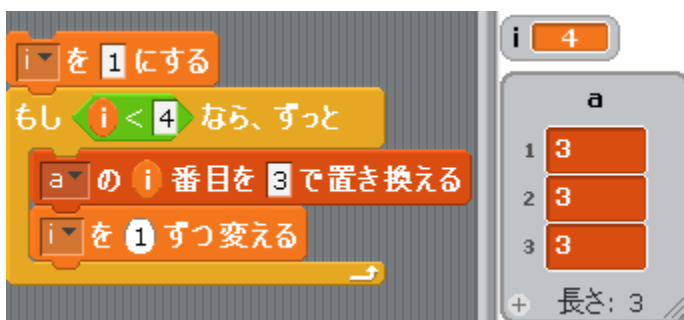


を使います。この方が、複雑なプログラムを書くとき書きやすいからです。

例えば、10回繰り返すには、



これを使うと、前節の例1は次のようになります。



また、例5の最大値を求めるプログラムは次のようになります。

The image shows a Scratch script for finding the maximum value in an array. The script starts by setting 'max' to the first element of array 'a' (index 1). Then, it enters a loop that continues as long as 'i' is less than 4. Inside the loop, it checks if the element at index 'i' of array 'a' is greater than 'max'. If so, it updates 'max' to that element. Finally, it increments 'i' by 1 and repeats the loop.

```
max を a の 1 番目 にする
i を 1 にする
もし i < 4 なら、ずっと
  もし a の i 番目 > max なら
    max を a の i 番目 にする
  i を 1 ずつ変える
```

On the right side, there is a variable monitor for 'i' with the value 4. Below it is a monitor for array 'a' with the following values:

Index	Value
1	2
2	7
3	4

Below the array monitor, it says '+ 長さ: 3'. At the bottom, there is a monitor for 'max' with the value 7.

単なる、置き換えですので慣れてしまえば簡単です。後は、Java プログラムにそのまま移行できます。

## 8. スプライト間の通信-ブロードキャスト（放送） -

スプライト間、やスプライトとステージ間でメッセージを送りあうには、「ブロードキャスト」という機能を使います。ブロードキャストとは放送のことで、ブロードキャストでメッセージを送ると、そのメッセージは全てのスプライトとステージのスク립トに飛んでいきます。

これに使用する命令は次の3つです。

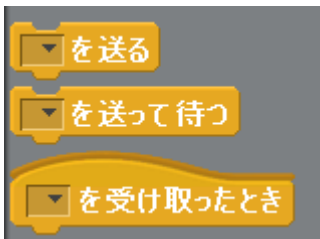
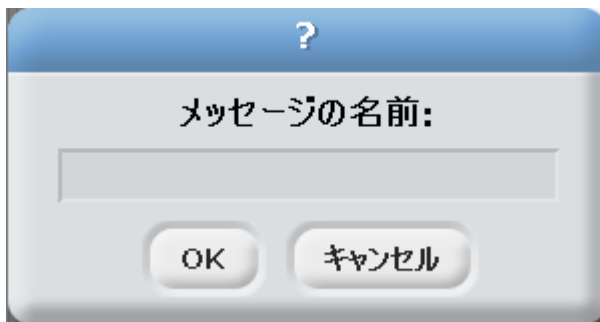


図8. 1 ブロードキャストに使う命令

ブロードキャストを行うには、先ず、送るメッセージを作る必要があります。メッセージを作るには上のいずれかの命令の□▼部の▼をクリックすると「新規」という吹き出しが出るので、吹き出しをクリックすると次の入力ボックスが現れます。



ここにメッセージ名を書き込んで、OKボタンを押すと、メッセージができ、送信や受信が可能になります。

### 8. 1 「～を送る」命令を使う

「を送る」命令で、スプライト間の情報交換ができます。

ここでは、シューティングゲームでよく使う技を示します。

猫は↑キーで上へ動き、↓キーで下に動く。スペースキーを押すと、猫から怪獣が飛び出します。

スペースキーが押されると、「shoot」という信号を送ります。



(猫のスク립ト)

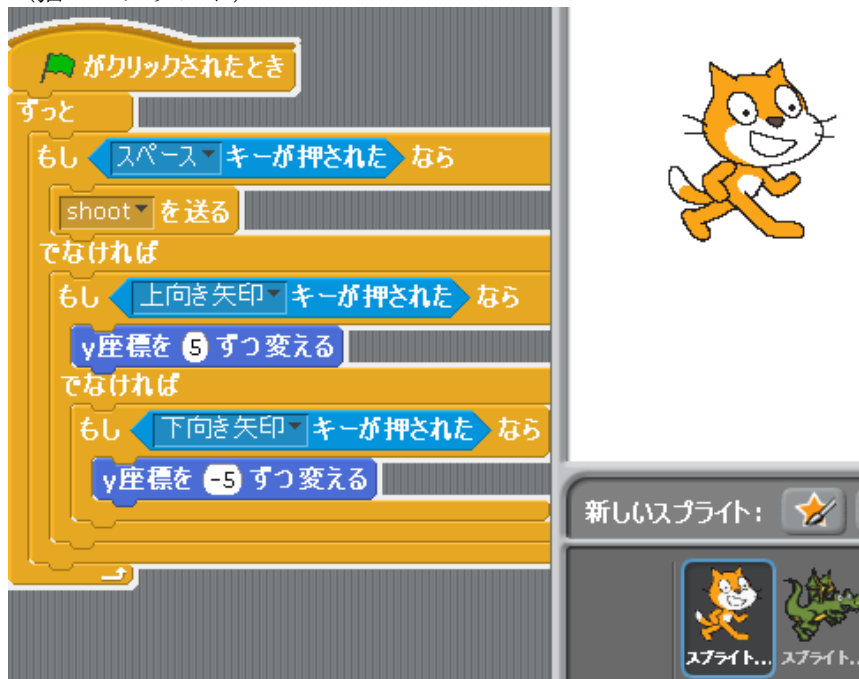


図8. 2 猫のプログラム

(怪獣のスク립ト)

怪獣スプライトは、緑旗が押されると 50%の大きさに縮小され、姿を消す。怪獣は Shoot という信号を受けると、姿を消したまま猫の後ろに移動し、その後画面に現れ、右に動き、途中で消える。

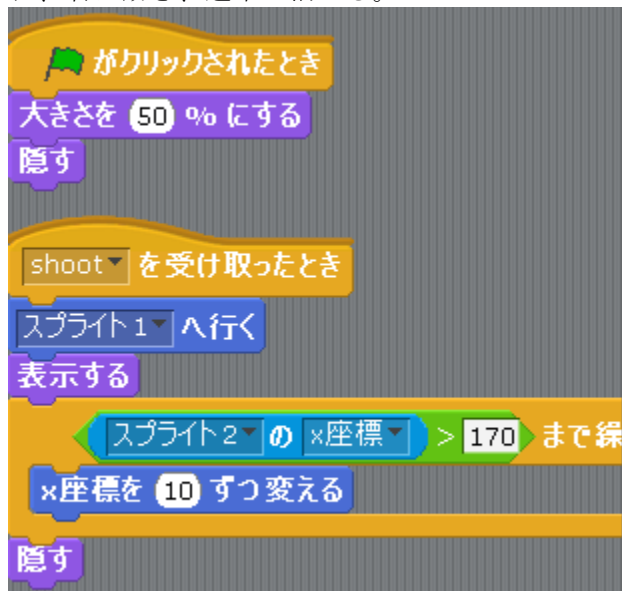


図8. 3 怪獣のスク립ト

## 8. 2 画面切換えの技

(1) スタート画面や終了画面なども1つのスプライトにすることができます。



図 8. 4 スタート画面

スタート画面は、SCRATCH に付いているペイントエディターで作れます。

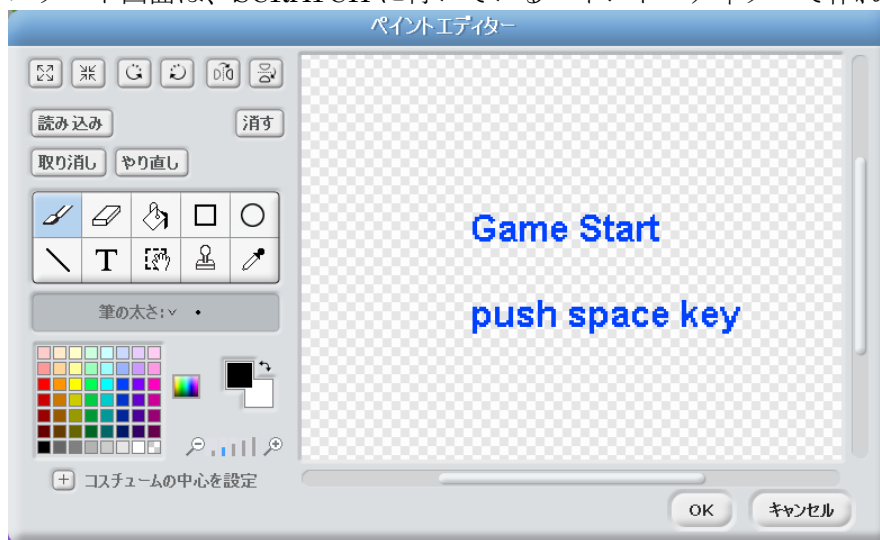


図 8. 5 描画環境

- (2) ステージは下図の右下の「ステージ」をクリックすると、スクリプトが書けて、背景の切り換えを行える。

下は、stra メッセージ（信号）が来ると背景が”desrt”に変わることになる。



図 8. 6 ステージのスク립ト

背景を複数用意しておいて、スプライトからメッセージを送ってステージのスク립トでメッセージを受け、背景を変えることもできます。

### 8. 3 スプライトから背景変更指令を送る例

2秒ごとに a と b というメッセージをブロードキャストし、ステージの背景を変えています。



図 8. 7 背景の切り替え

## 9. その他の便利な機能

(1) スプライトをファイルから選ぶと、予め「障害物」などのスクリプトの付いているスプライトもあります。



図9. 1 スクリプト付きオブジェクト

(2) 他のスプライトに属している変数へのアクセスも可能

例えば、猫がバナナを買ったとして、バナナの値段(nedan)を猫の財布から引く場合、次のようにする。



図9. 2 猫のスクリプト

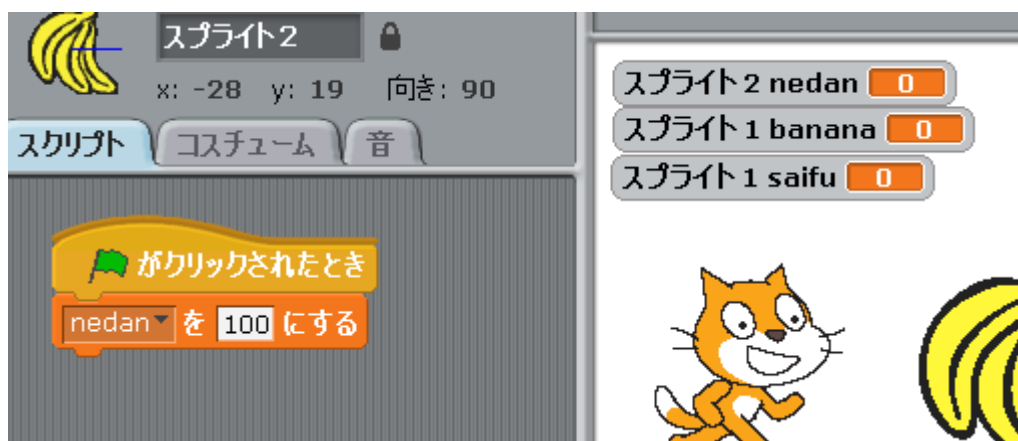


図9. 3 バナナのスク립ト

### (3) リスト

7章では、「リストを作る」というボタンで配列を作りましたが、リストを作ると、キュー(FIFO)やスタック(LIFO)ができます。(とりあえず、意味不明でOK)

### (4) 作品の保存

作品の保存には、「ファイル」->「名前をつけて保存」を指定し、ファイル名を指定。

ファイルの拡張子は **sb** です。