

## もくじ 目次

I. <b>初級編</b> <small>しょきゅうへん</small>	1
1. フロック <b>崩</b> <small>くず</small> しゲーム.....	1
2. ラケット(プレイヤー)を <b>作成</b> <small>さくせい</small> しよう.....	2
3. プレイヤーを <b>動</b> <small>うご</small> かそう.....	4
4. フロックを <b>作成</b> <small>さくせい</small> しよう.....	7
5. ボールを <b>作成</b> <small>さくせい</small> しよう.....	8
6. ボールに <b>動</b> <small>うご</small> きをつけよう.....	9
7. フロックを <b>消</b> <small>け</small> してみよう.....	11
8. ゲームオーバー.....	12
9. <b>旗</b> <small>はた</small> を <b>押</b> <small>お</small> したらプレイできるようにしよう.....	14
10. フロックを <b>増</b> <small>ふ</small> やしてみよう.....	16
II. <b>中級編</b> <small>ちゅうきゅうへん</small>	23
1. <b>初級編</b> <small>しょきゅうへん</small> のおさらい.....	23
2. <b>他</b> <small>ほか</small> のスプライトにメッセージを <b>送</b> <small>おく</small> ってみよう.....	24

3.	<sup>へんすう</sup> <b>変数</b> を使ってみよう	27
4.	<sup>のこ</sup> <b>残</b> っているブロックをカウントしてみよう	31
5.	クリアした <sup>とき</sup> <b>時</b> にボールの <sup>うご</sup> <b>動き</b> を <sup>と</sup> <b>止</b> めよう	33
6.	<sup>おんせい</sup> <b>音声</b> を <sup>い</sup> <b>入</b> れてみよう	37
7.	ステージを <sup>つく</sup> <b>作</b> ろう	45
8.	ステージの <sup>はいけい</sup> <b>背景</b> を <sup>か</sup> <b>変</b> えてみよう	48
9.	ステージごとにブロックの <sup>かず</sup> <b>数</b> を <sup>か</sup> <b>変</b> えよう	54
10.	ステージごとにボールの <sup>はや</sup> <b>速</b> さを <sup>か</sup> <b>変</b> えよう	60
11.	<sup>しょうがいぶつ</sup> <b>障害物</b> を <sup>つく</sup> <b>作</b> ってみよう	64
III.	<sup>じょうきゅうへん</sup> <b>上級編</b>	75
1.	ゲームの <sup>なが</sup> <b>流</b> れを <sup>つく</sup> <b>作</b> ってみよう	75
2.	ゲーム <sup>しゅうりょう</sup> <b>終了</b> の <sup>とき</sup> <b>時</b> に <sup>しょうがいぶつ</sup> <b>障害物</b> を <sup>け</sup> <b>消</b> してみよう	86
3.	ステージを <sup>ふ</sup> <b>増</b> やしてみよう	88
4.	<sup>しょうがいぶつ</sup> <b>障害物</b> を <sup>ふ</sup> <b>増</b> やしてみよう	96
5.	<sup>はいけい</sup> <b>背景</b> にアニメーションを <sup>い</sup> <b>入</b> れてみよう	102
6.	<sup>すべ</sup> <b>全</b> てのステージのクリア <sup>えんしゅつ</sup> <b>演出</b> を <sup>い</sup> <b>入</b> れてみよう	107

IV.もしもうまく**動作**しない**場合**..... 114

1. **当**てた**時**に「ボール」がその**場**で**振動**する**場合**..... 115

2. **次**のステージに**進**んだ**時**、フロックが1つない**場合**..... 121

目次

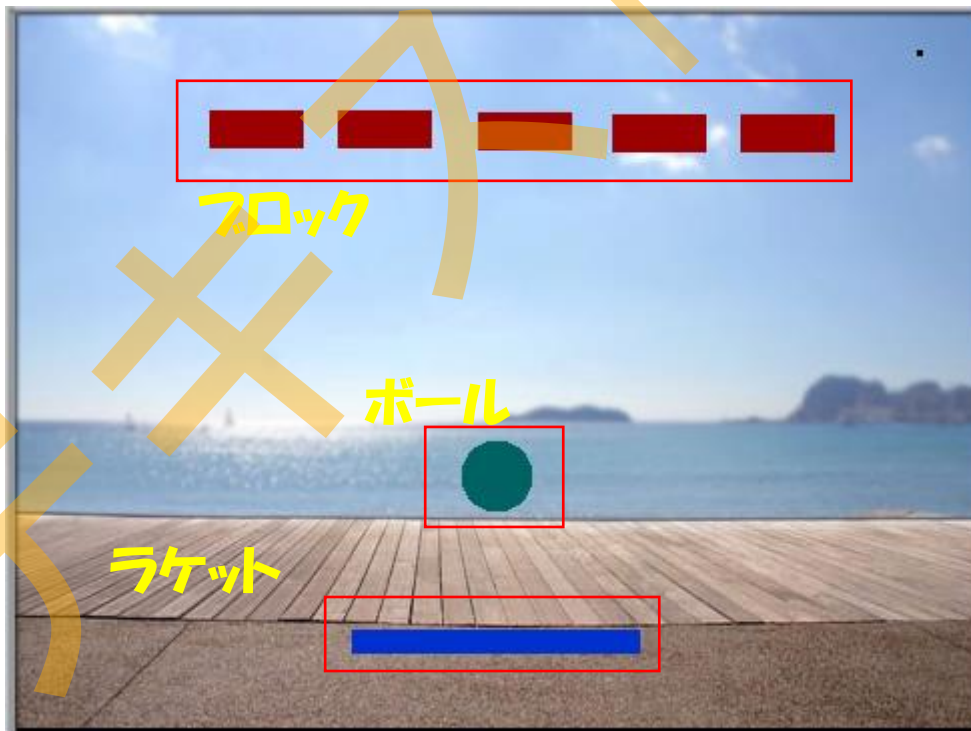
# しょきゅうへん I. 初級編

## 1. フロック崩しゲーム

まず、最初にゲームの説明をします。

今回作成するゲームは下の図のようにステージに置かれているブロックをボールで当てて消していくゲームです。ステージの下にはラケットがあり、落ちてくるボールをラケットで打ち返して、一番下に落ちないようにします。

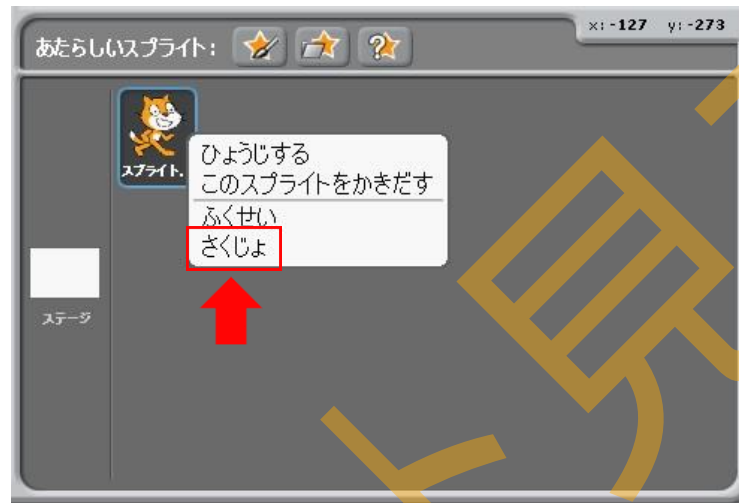
ゲームを開始するとブロックの下にあるボールがラケットに向かって落ちてきます。プレイヤーはラケットを左右移動させて、上方にあるブロックに向かって落ちてきたボールを打ち返します。



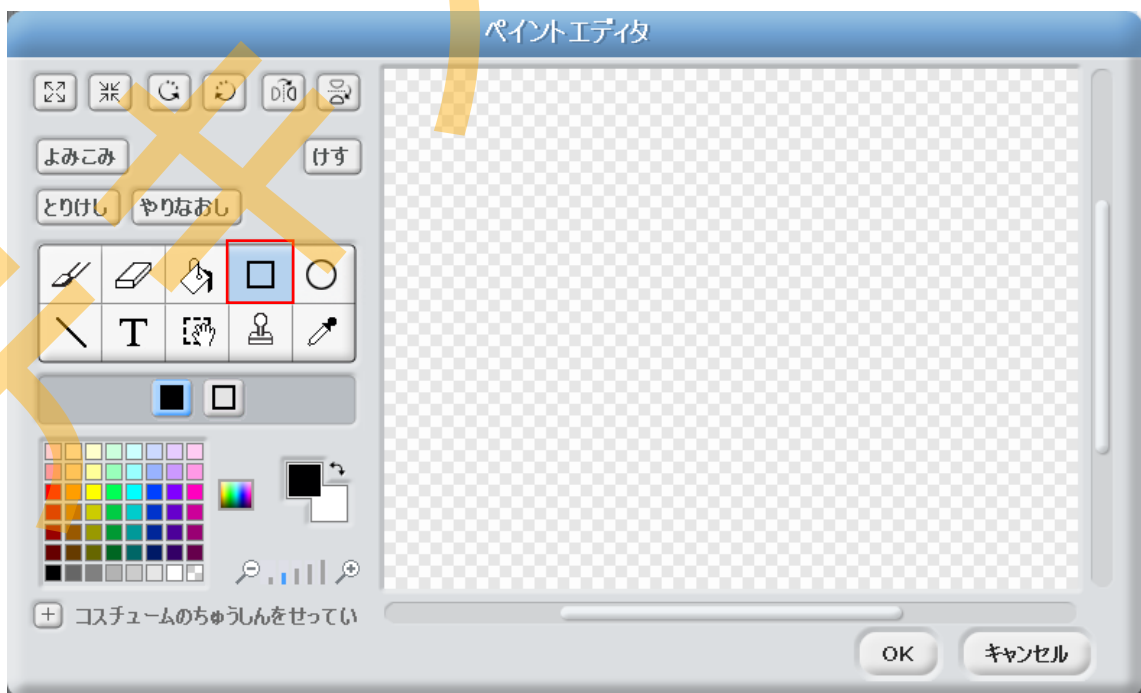
## 2. ラケット(プレイヤー)を<sup>さくせい</sup>作成しよう

この作業をする前に今回は使用しない猫の絵を消してしまいましょう。

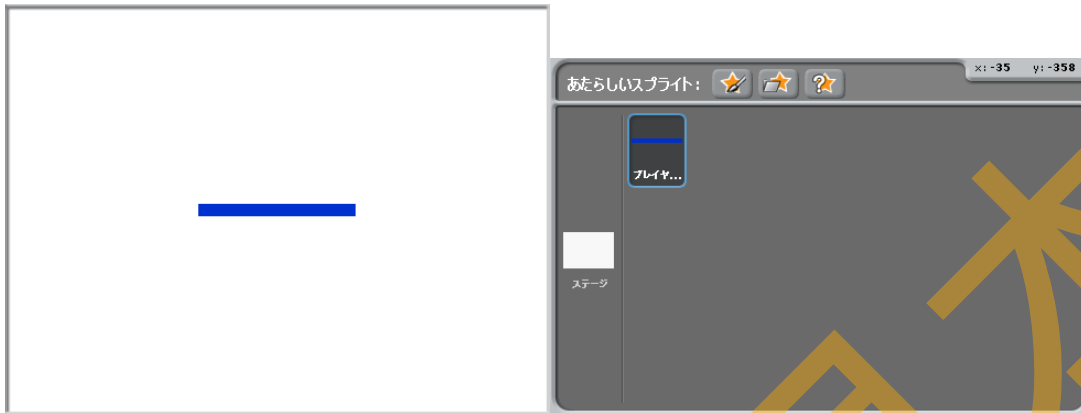
スプライトを<sup>みぎ</sup>右クリックし、『さくじょ』をクリックします。



それではプレイヤーとなるラケットを<sup>さくせい</sup>作成しましょう。『あたらしいスプライトをかく』でラケットを<sup>さくせい</sup>作成します。ラケットは<sup>しかく</sup>四角い形<sup>かたち</sup>をしています。



四角を描くには  を使うと綺麗に描くことができます。



描いた絵（スプライト）が表示されていれば完了です。

スプライトの名前は「プレイヤー」にしておきます。

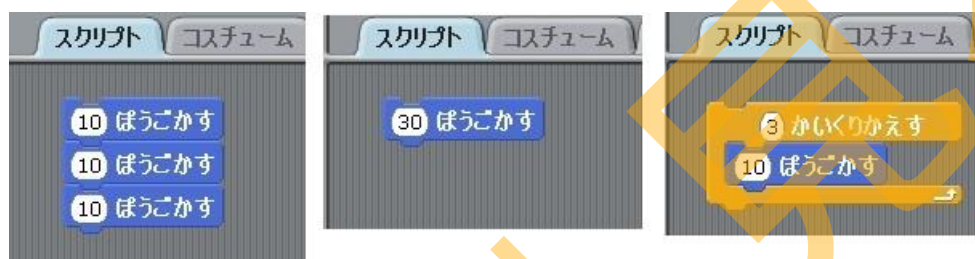
スクリプトエリアの名前をクリックして「プレイヤー」と入力しましょう。



### 3. フレイヤーを動かそう

早速プレイヤーを動かしてみましょう。

今回は←キーを押したとき左に30歩、→キーを押したとき右に30歩動くようにします。



1 動かし方には色々な方法がありましたね。

どれを使用しても構いませんが、ここでは中央の「30歩動かす」で動かすことにします。スクリプトに「10歩動かす」を追加し、「30歩動かす」に変えましょう。

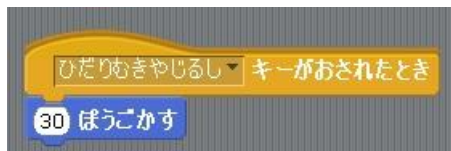
2 次は左右に動かせるようにします。

「せきぎょ」の中から「スペース キーがおされたとき」を挿入して、「スペース」を「ひだりむきやじるし」に変えておきます。同じように『みぎむきやじるしが おされたとき』も追加します。



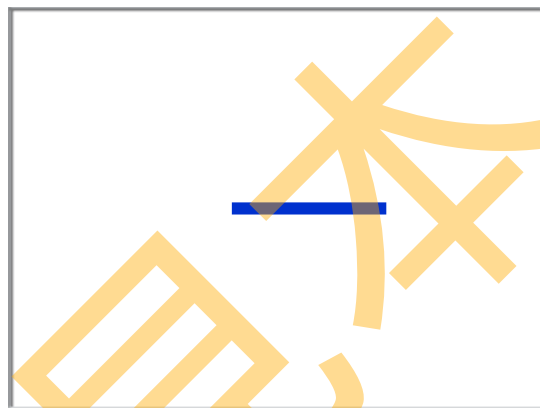
3 スクリプトエリアの  に  をつなげ

て、キーボードの←キーでプレイヤーを左に動かしてみましよう。



あれっ? ←キーを押したのに右に移動しましたね。

これはプレイヤーの向きが90度(みぎ)になっているからです。



? では、左に動かすにはどうすればいいでしょう?



## ◇問題の答え◇



のままでは向きを変えないで30歩動いてしまう

のでプレイヤーに「左向きに動く」という命令を出さないといけません。

『うごき』の中から『90 どのむける』を『キーがおされたとき』と『30 ぼうごかす』の間に入れます。

左に動かしたいので『90 どのむける』を、  
-90度（ひだり）にすれば向きを左に変えた  
後30歩動かすので左に移動できます。

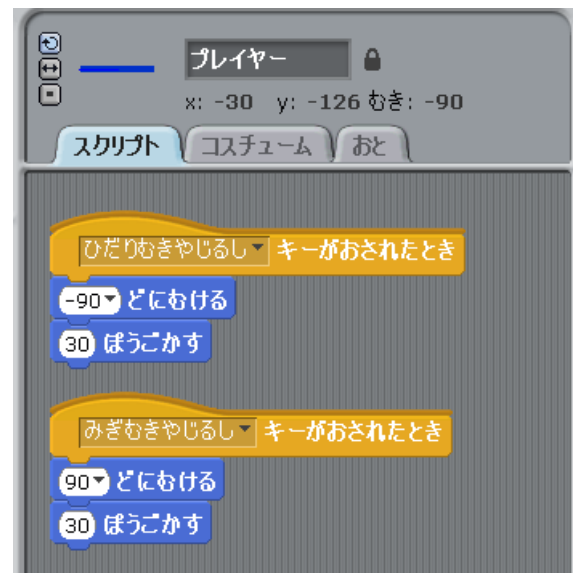


これで左のキーを押すとプレイヤーは左に移動するようになりますが、右のキーを押すとプレイヤーは左に動いてしまいます。

右方向に動かすには今度は「右向きに動く」という命令を出します。

右に動かしたい場合は『90 どのむける』を『キーがおされたとき』と『30 ぼうごかす』の間に入ればよいです。


右の図のようになっていれば完了です。

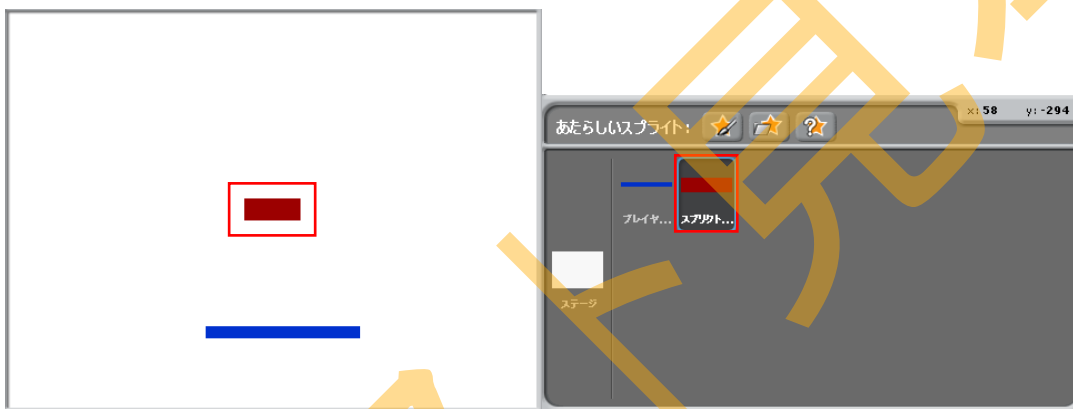


## 4. ブロックを作成しよう

つぎにブロックを作成しましょう。

今回は5つのブロックを用意します。まずは1つだけ作成してみましょう。

ブロックはプレイヤー（ラケット）と同じ四角の形をしていますのでペイントエディタの  を使って描きます。



ブロックに対するプログラムは後に説明します。


ここではまだブロックを1つ作成するだけにしておきましょう。

スプライトの名前は「ブロック 1」にします。

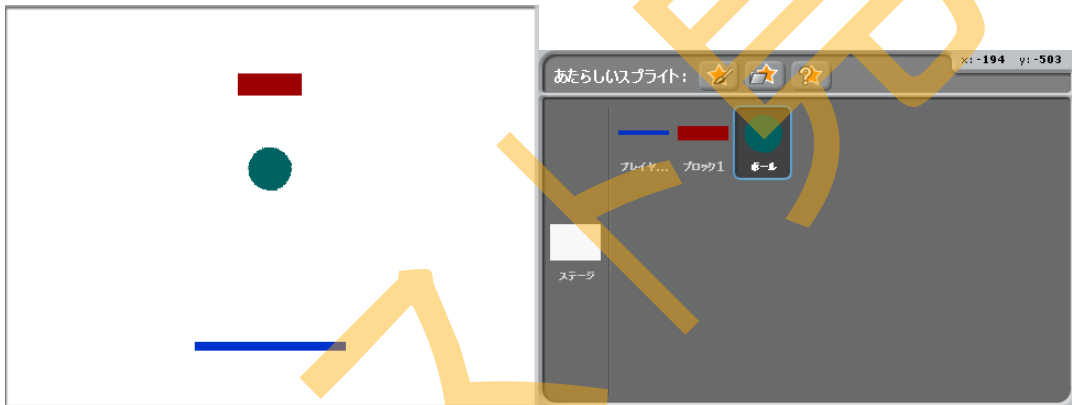


## 5. ボール<sup>さくせい</sup>を作成しよう

次にボール<sup>さくせい</sup>を作成します。

ボール<sup>まる</sup>は丸<sup>かたち</sup>の形<sup>かたち</sup>をしていますから  を使用<sup>しよう</sup>します。

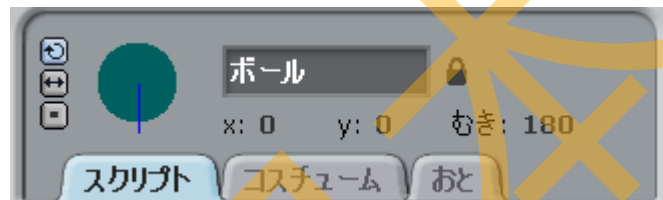
スプライトの名前<sup>なまえ</sup>は「ボール」にします。



## 6. ボールに動きをつけよう

ブロック崩しは、ボールは常に動いていて壁や、ラケット、ブロックに当たることによって進行方向を変えます。

最初にスプライトエリアでスプライトの名前や形を確認してからパーツを挿入していきましょう。スプライトがいくつもある場合はこの確認を忘れないようにしましょう。

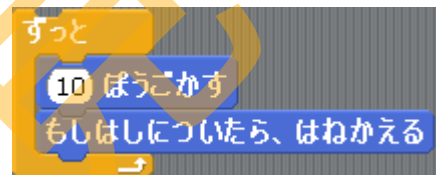


ボールのスクリプトにパーツを挿入していきます。

常に動くようにするためには「せいきよ」にある



→を使います。



ここに「10 ぼうごかす」と「もしはしについたら、はねかえる」を挿入すると壁に当たるとバウンドするようなプログラムになります。「10 ぼうごかす」では速いかもしれないので数字を3に変えましょう。「もしはしについたら、はねかえる」は上下左右のどの端でも跳ね返ります。スプライトの向きを変えて動かして試してみましょう。

次にラケットとブロックに当たった場合です。

この場合ではボールの進行方向を変えないとゲームとして面白味がありません。

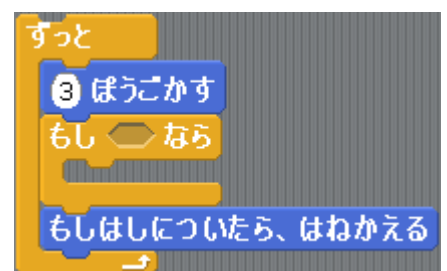
そこでもラケットに触れた時や、ブロックに触れた時はボールの進行方向を変えてみましょう。

1 まずはどのパーツを使用するのかを考えます。

今回はラケットに触れた時やブロックに触れた時の条件でボールの進行方向を変えます。条件の中で動きを命令

できるのは「せいきよ」の「もし」なら

これを右の図のように挿入します。



**2** 次にラケットに触れた場合やブロックに触れた場合について考えます。

しらべる の中に にふれた があります。

```

ずっと
  ③ ぼうごかす
  もし にふれた なら
  もしはしについたら、はねかえる
  
```



これを に挿入します。

次に にふれた を「プレイヤーにふれた」に変更します。右の からプレイヤーを選択します。

```

ずっと
  ③ ぼうごかす
  もし マウスのポインターはし
  ブロック1
  プレイヤー
  
```

**3** この中に進行方向を変える命令を挿入します。

ここでは 90 どのむける のように一定の方向にするの

ではなく 15 どのまわす を使います。

これを挿入して数字を 80 にしてみましょう。

```

10 ぼうごかす
15 どのまわす
15 どのまわす
90 どのむける
  
```

ブロックが触れた時も同じような命令を出してみよう。

```

がクリックされたとき
ずっと
  ③ ぼうごかす
  もし プレイヤー にふれた なら
  80 どのまわす
  もし ブロック1 にふれた なら
  80 どのまわす
  もしはしについたら、はねかえる
  
```

# 1. ブロックを消してみよう

それではボールがブロックに当たった時にブロックを消すようなプログラムを作成しましょう。

ここで言うことはスプライトをステージから削除することではなく、見えないようにすることです。

**みため**の中には**かくす**と**ひょうじする**があります。

**かくす**は見えているスプライトをステージから見えないようにします。

**ひょうじする**は見えなくなっているスプライトを見えるようにします。

これを使ってボールが当たった時にブロックを消してみましよう。

これはブロックのスプライトに対して行います。



スプライトエリアからブロック1のスプライトを選んでクリックします。

**1** ブロックはボールが当たった時に消えますが、ボールが当たるまでブロックを消すという命令を待たなくてはなりません。

**せいぎよ**の中にある**までまつ**のパーツを使います。

これをブロックのスクリプトに挿入します。

条件はボールに触れた時ですから**ボールにふれた** **までまつ**になります。

**2** **ボールにふれた** **までまつ**の後にボールに触れた時の命令を出します。

ボールに触れた後、ブロックを消すために

**かくす**のパーツを使います。

これでボールが当たるとブロックが消えます。



## 8. ゲームオーバー

ブロック崩しはボールが下に落ちてしまうとゲームオーバーになってしまいます。

ゲームオーバーになった時はボールのエフェクトを変えてみましょう。

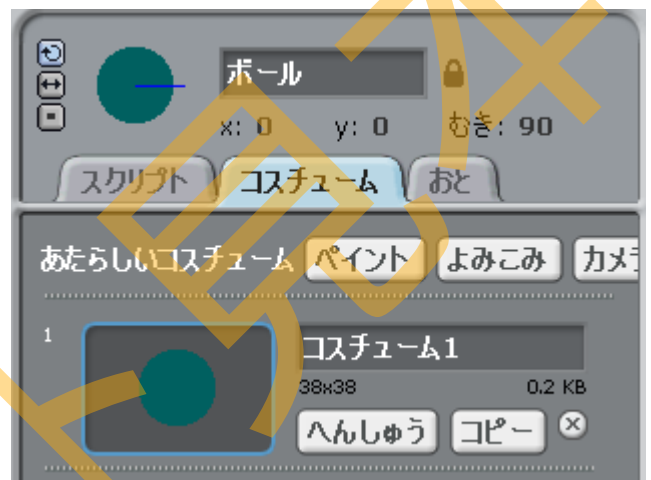
そのために、まず、ボールに新たなコスチュームを追加しましょう。

ボールのスプリクトエリアのコスチュームから「あたらしいコスチューム」で新しいコスチュームを追加します。

下のようなコスチュームを追加します。



これはコスチューム2になります。



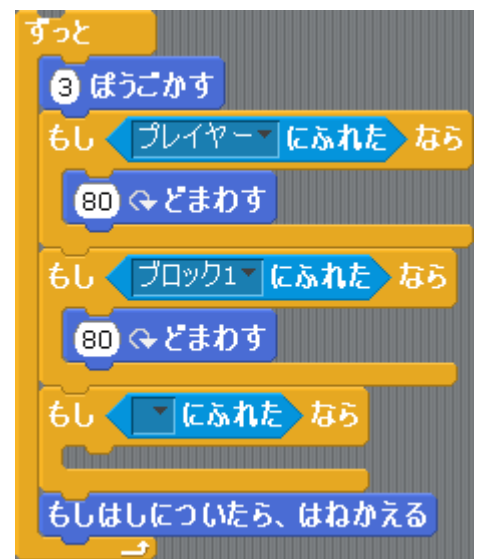
それではゲームオーバーのプログラムを作成してみましょう。


1 下の端に触れた時にゲームオーバーにするので、

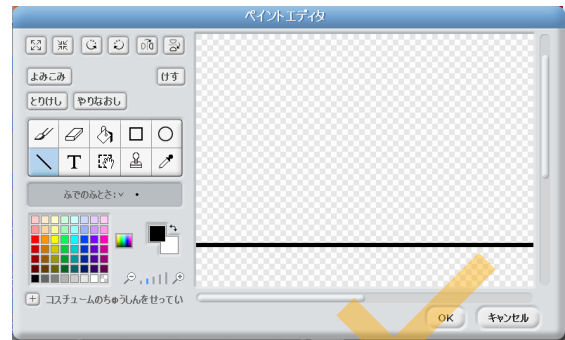
もし なら と の挿入はプレイヤーに触れた時と同じです。

しかし、下の端だからといって にしてしまおうと端なら左でも右でも上でも触れた場合に命令を実行してゲームオーバーになってしまいます。

そこで横に細長いスプライト作成しましょう。

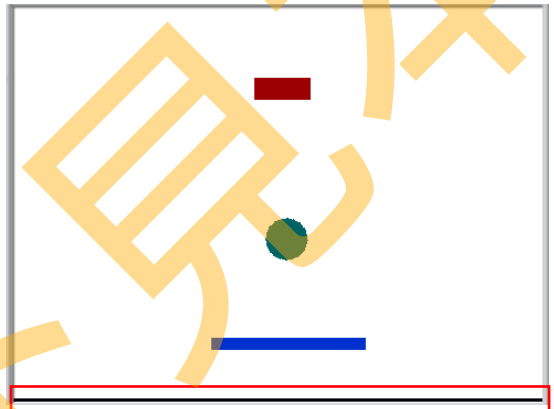


ペイントエディタで  を使えば簡単に線が描けます。



**2** スプライトを作成したら下の端に移動します。

スプライトの名前は「ミス」にしておきます。



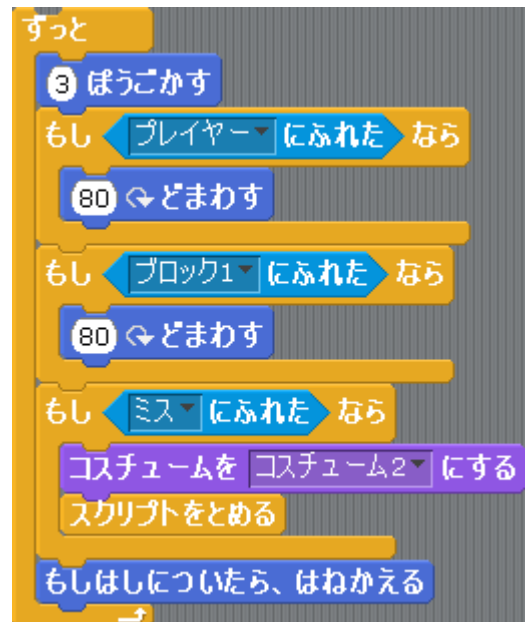
**3** それでは再びボールのスク립トに戻りましょう。



にしてコスチュームを変える命令を挿入します。



またゲームオーバーになるのでボールを止めなくてはなりません。コスチュームは **みため** にある **コスチュームを コスチューム2 にする** で変えましょう。ボールの動きを止めるには **せいぎよ** にある **スクリプトをとめる** を使います。

右の図のように挿入しましょう。





## 9. 旗を押したらプレイできるようにしよう

ステージにあるを押したらプログラムが動くようにする命令は、**せいぎよ**にあるです。

これを押した時にプレイヤー、ボール、ブロックには以下のようなことをしておくてはなりません。

プレイヤー

1. ゲーム開始の時に決まった位置に設定する。

ボール

1. ゲーム開始の時に決まった位置に設定する。
2. ボールの向きを下に向けておく。
3. ゲームオーバーになった時のエフェクトをボールに戻す。

ブロック

1. 消えたブロックを元に戻す。

それぞれの場合にどのような命令を挿入していくかを説明しましょう。

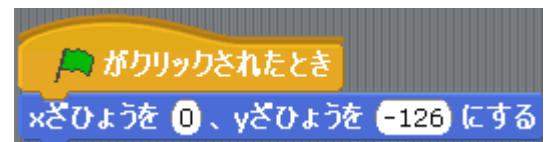
### プレイヤーの場合

**1**  **がクリックされたとき**を挿入します。

**2** 位置を設定するためには**うごき**にある **x座標を 0、y座標を -126 にする**を使います。

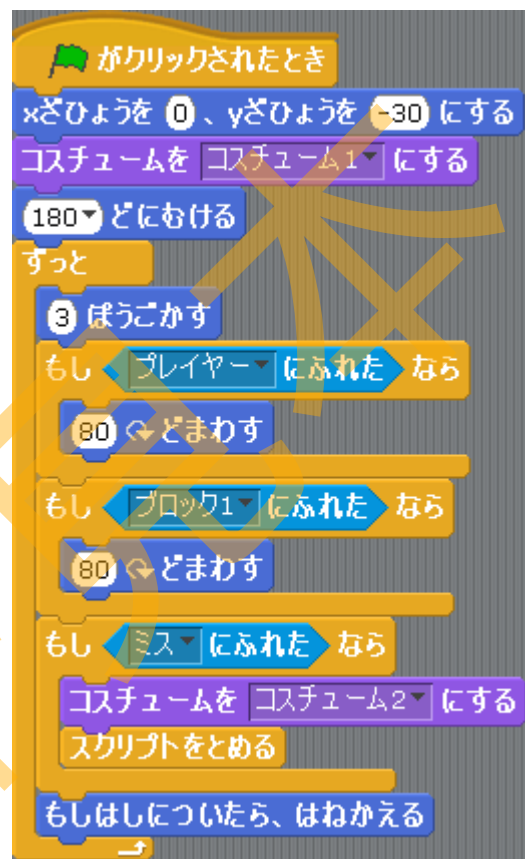
横は真ん中にしたいので x座標は 0 にしましょう。

このプログラムはキーを押す場合のプログラムにつなげないようにしましょう。



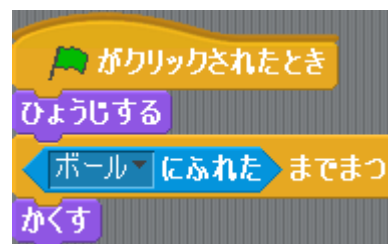
## ボールの場合

- 1 **がクリックされたとき** を挿入します。
- 2 位置を設定します。プレイヤーよりも高い位置に設定しておきましょう。  
例として **x座ひょうを 0、y座ひょうを -30 にする** のように設定しました。これを挿入します。
- 3 スプライトをボールの形に戻しておきます。ボールはコスチューム 1 なので **みため** にある **コスチュームを コスチューム1 にする** でボールに戻ります。
- 4 **180 どのむける** を挿入してボールの動きのプログラムと繋ぎ合わせます。



## ブロックの場合

- 1 **がクリックされたとき** を挿入します。
- 2 消えたブロックを元に戻すということはブロックを見えるようにします。  
**みため** から **ひょうじする** をつなげます。
- 3 ブロックが消えるプログラムをつなげていきます。



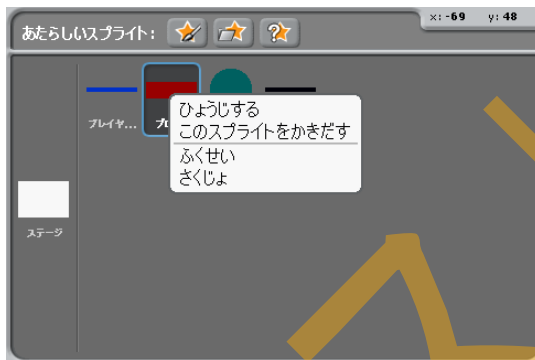
## 10. ブロックを増やしてみよう

ブロック崩しはゲーム画面に散らばっているいくつかのブロックを消していくゲームです。

したがって、今は1つしかないブロックを増やしてみましょう。

新しくスプライトを描いてもいいですがここではスプライトのコピーを使いましょう。

例として4つのブロックを増やして画面に5つのブロックがあるようにします。



スプライトエリアにあるスプライトを右クリックして「ふくせい」を選択します。



このように同じスプライトが追加されます。

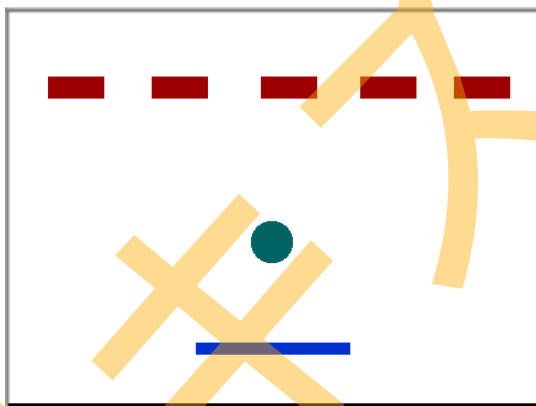
このスプライトのプログラムの内容を見ると、コピー元と全く同じであることがわかります。



同じようにブロックをあと3つ作成します。

作成ができたならブロックはスプライトをマウスでドラッグしてステージの上の方に横に並べておきましょう。

スプライトの名前は「ブロック1」、「ブロック2」、「ブロック3」、「ブロック4」、「ブロック5」にしておきます。



? マウスでスプライトを移動した場合、多少ずれたりして調整が大変です。簡単にかつ、綺麗に並べるにはパレットにある何のパーツを使えばいいでしょう？

## ◇問題の答え◇

図では高さ、つまり、y の座標を合わせていますから、y の座標を設定できるパーツを使えば調整が簡単になります。

そのパーツとは「うごき」の中にある「y座標を 0 にする」になります。

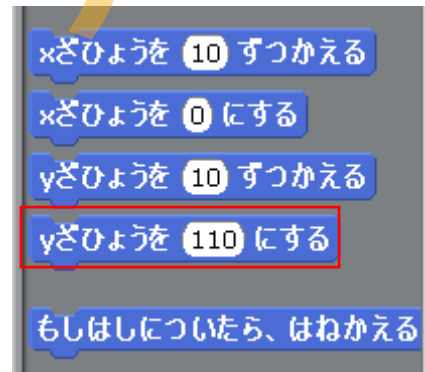
まず、1つ目のブロックの位置を決めます。スクリプトエリアでyの座標を確認することができます。

右の図ではyの座標は110になっていますので他のブロックもこの座標に合わせてみます。

他のブロックのSpriteを選択してブロックパレットにある「y座標を 0 にする」を「y座標を 110 にする」に変更してクリックします。

これで1つ目のブロックと同じ高さになります。

他のブロックも同じようにしてみましょう。



x座標も同じように「x座標を 0 にする」を使って指定することができます。そこで

各ブロックのx座標を以下のように指定しましょう。

ブロックの名前	指定するx座標
ブロック 1	-200
ブロック 2	-100
ブロック 3	0
ブロック 4	100
ブロック 5	200

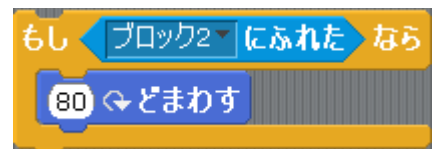
さて、コピーしたブロックのにはすでにプログラムがあり、同じ動きをすればいいのでプログラムを変更する必要はありません。

しかし、ボールのにはコピーしたに触れた場合のプログラムを追加しなくてははいけません。その方法を説明していきましょう。

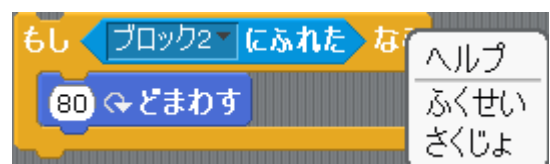
**1** ブロックに触れた時の命令は右の図の赤い枠になります。つまり、コピーしたブロックにも同じような命令を出します。まずは赤い枠と同じようなプログラムをエリアで作成しましょう。



**2** まずはブロック2が触れた場合のプログラムを作成しましょう。



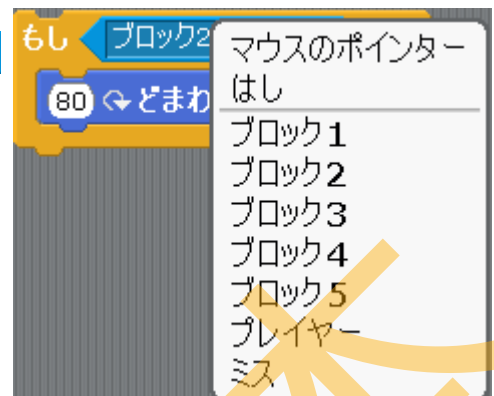
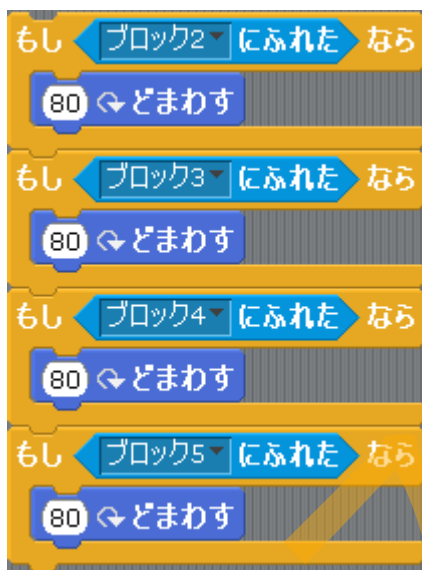
**3** 同じようにブロック3、ブロック4、ブロック5が触れた場合のプログラムを作成します。ブロック2が触れた場合のプログラムを右クリックして「ふくせい」からコピーします。



コピーした3つのプログラムの「ブロック2」にふれたをそれぞれ「ブロック3」、「ブロック4」、ブロック5」に変更します。

これでブロック3、ブロック4、ブロック5にふれた時のプログラムを作成しました。

これらのプログラムをつなげます。




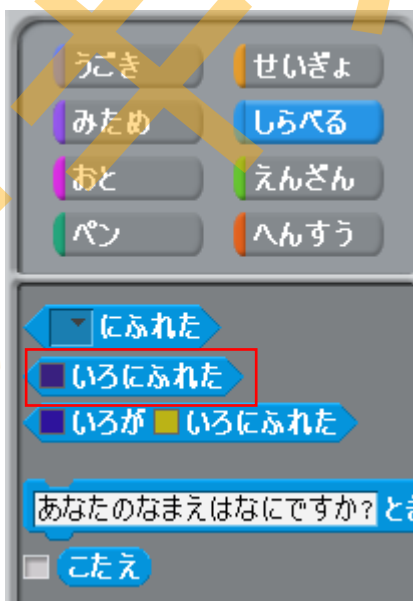
**4** このプログラムをボールの動きのプログラムに挿入します。

つぎのページの右の図のようになっていれば完成です。

? <sup>みぎ</sup> <sup>ず</sup> 右の図ではプログラ  
<sup>なが</sup> ムが長くなっています。  
<sup>した</sup> <sup>あか</sup> <sup>わく</sup> <sup>かこ</sup> 下の（赤い枠で囲んで  
 いる）パーツを使っ<sup>つか</sup>てプ  
 ログラムを短<sup>みじか</sup>くしてみ  
 しょう

■ **いろにふれた** <sup>いろ</sup> <sup>せっていほうほう</sup> の色の設定方法を  
<sup>せつめい</sup> 説明します。

■ **いろにふれた** の  をクリックす  
 るとマウスが  の形<sup>かたち</sup>に変わります。  
 この状<sup>じょうたい</sup>態でクリックすると  がク  
 リックした場<sup>ばしょ</sup>所の色<sup>いろ</sup>に変わります。  
 ブロックの色<sup>いろ</sup>を設定<sup>せってい</sup>したい場<sup>ばあい</sup>合はス  
 テージのブロックをクリックします。





## ◇問題の答え◇

「**いろにふれた**」とは指定された色に触れた場合という意味です。

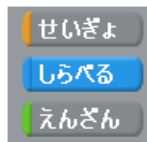
つまり、指定された色であればどんなスプライトでもよいのです。

ここでステージを見てみましょう。

ブロックを見ると全て同じ色になっています。

そして、他のスプライトにこの色はありません。

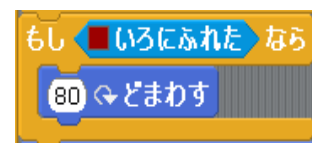
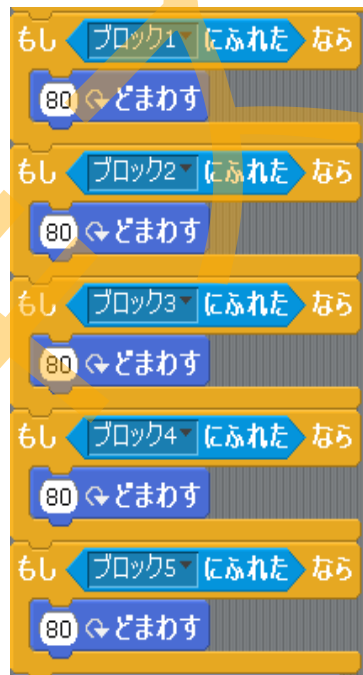
そして、各ブロックがボールに触れた時の命令内容は全て同じです。



「しらべる」から「いろにふれた」を選び、色を青に変えて

「**いろにふれた**」にすればブロック全てに対して命令が出せます。

下の図のように  
変えます。

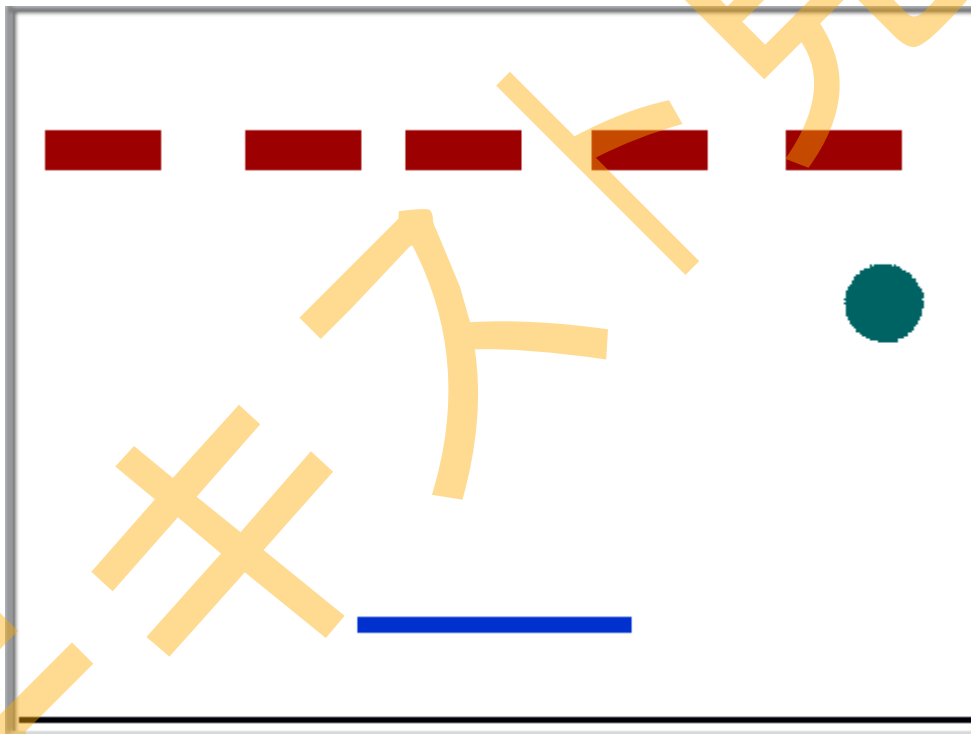


これでプログラムは短くなります。

## ちゅうきゅうへん II. 中級編

### しょきゅうへん 1. 初級編のおさらい

初級編ではプレイヤーの操作、ボールの動き、ボールが当たった時のブロックの消滅、ボールが下に落ちた時のゲームオーバーになるというところまでやりました。ここからはスプライトにメッセージを送ってみたり、全てのブロックを消した時にボールを止めるようにしたり、いろいろなシチュエーションで音声を鳴らしてみたりします。



初級編で作ったプログラムでは、「プレイヤー」と「ボール」「ミス」「ブロック」×5があるはずです。

## ほか 2. 他のスプライトにメッセージを送ってみよう

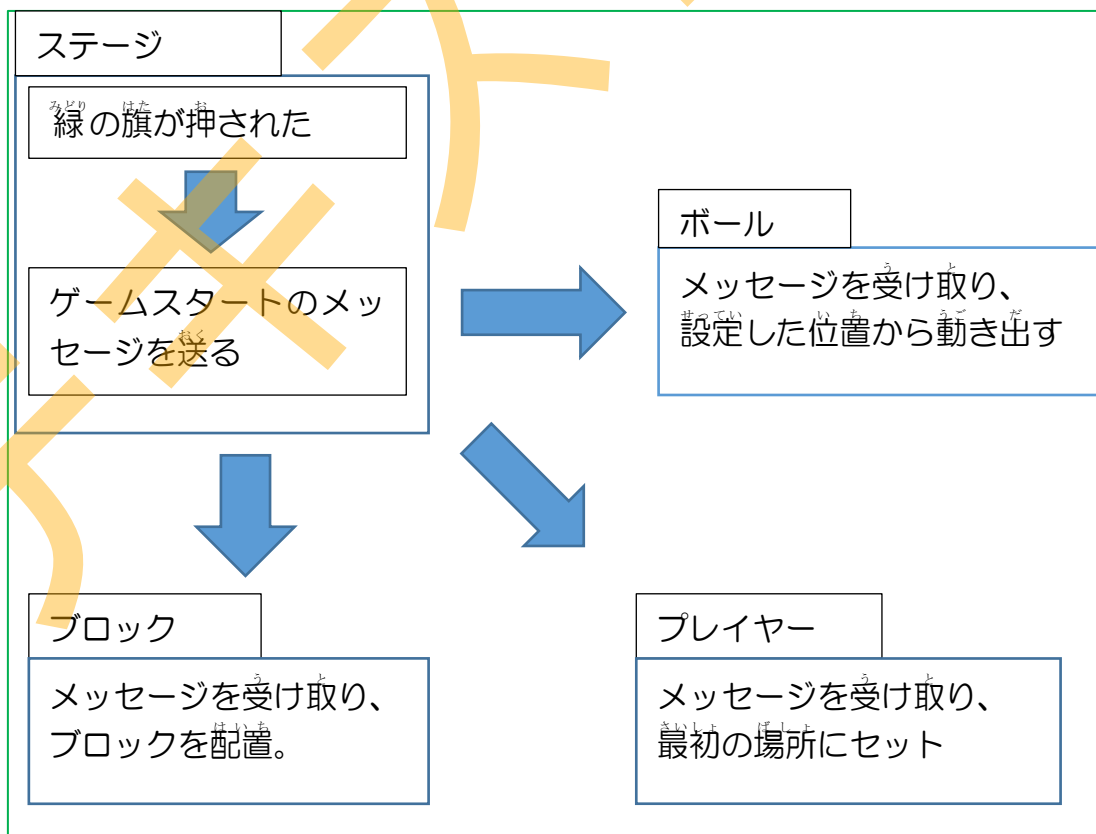
初級編のブロック崩しは  がクリックされたとき をクリックすれば動くようになっています。

これはボール、プレイヤー、ブロックのそれぞれに対して緑の旗をクリックすれば動くようにプログラムされているからです。

これを緑の旗が押されたら「ゲームが始まります。ボール、プレイヤー、ブロックはそれぞれの動きをしてください」というような感じのメッセージを送って、メッセージを受け取ったボール、プレイヤー、ブロックが動くようにしてみましょう。



ボール、プレイヤー、ブロックは全てステージの上にありますからステージからメッセージを送るようにしましょう。



この時に使うのが「せいぎょ」にある「おくる」と「うけとったとき」です。

この中にメッセージを自由に入力することができます。  
実際にこれを挿入してみましょう。



**1** スプライトエリアからステージを選択して、緑の旗をクリックした場合のプログラムを挿入します。

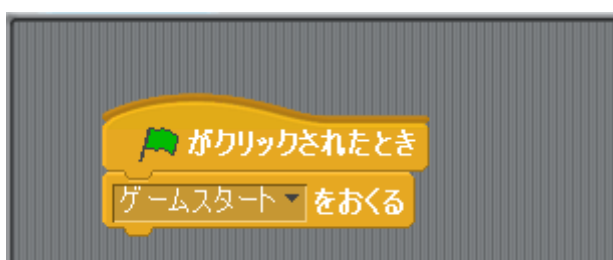
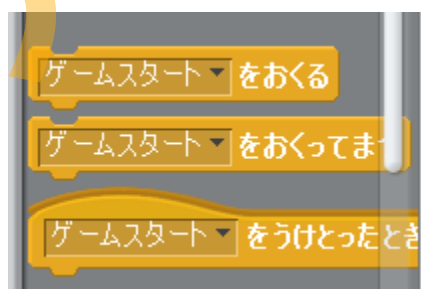


**2** では、「おくる」プログラムを挿入します。  
「おくる」の▼を選択します。  
次に「しんき」で送るメッセージを設定します。



名前は「ゲームスタート」にします。

ブロックパレットの「せいぎょ」をクリックすると「おくる」、「おくってまつ」、「うけとったとき」の空白がゲームスタートに変わります。



### 3 ボールのスクリプトで

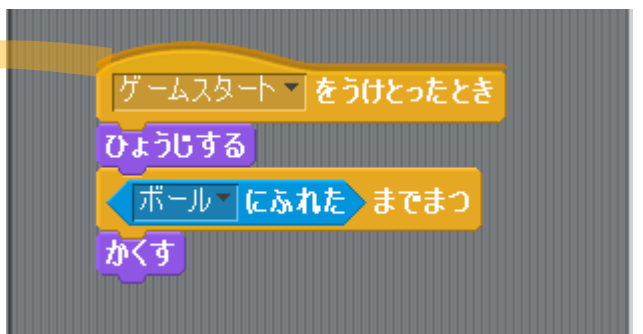
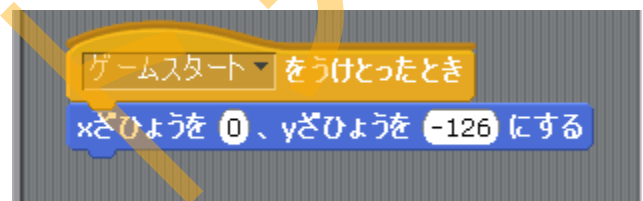
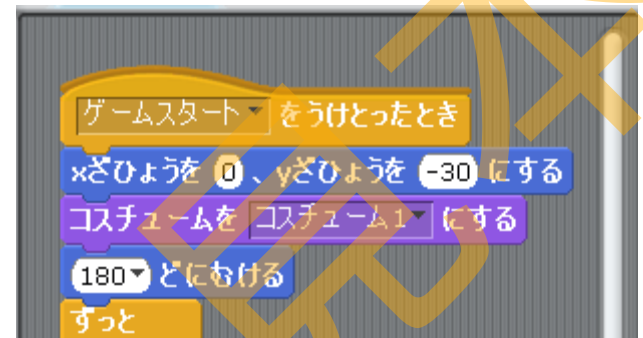
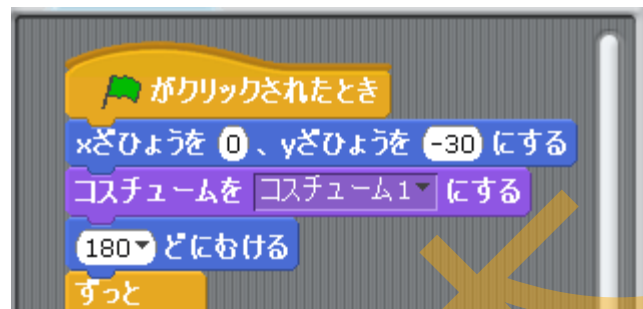
緑の旗がクリックされた時のパーツを削除して「ゲームスタートをうけとったとき」のパーツを代わりに置き換えます。

これでボールはステージからメッセージを受け取ったことになります。

プレイヤーとブロック1~5も同じように変更します。

さて、実行してみましょう。

このようにメッセージを送ることで、別のスプライトに動きを指示することができます。



### 3. 変数を使ってみよう

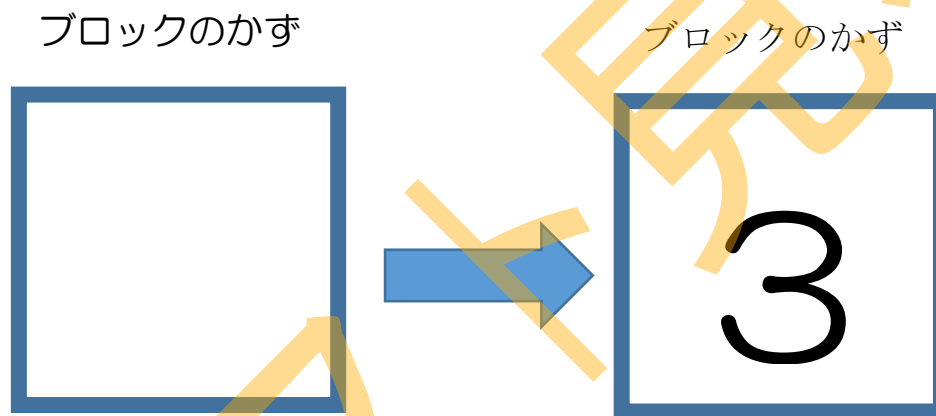
全てのブロックを消すと次のステージに移動するにはどうすればいいでしょう？

それには消されたブロックを数える必要があります。

今、何個のブロックが残っているかを入れておく箱が変数です。

箱には名前をつけることができ、それが変数名になります。

例えば、名前を「ブロックのかず」にします。



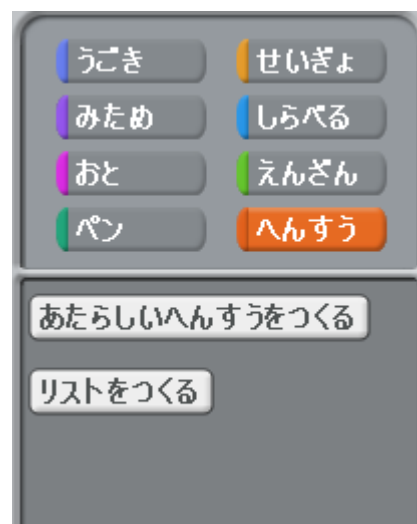
3を入れると……

それでは変数の使い方を説明していきましょう

**1** 変数を設定するにはブロックパレットの「へんすう」をクリックします。

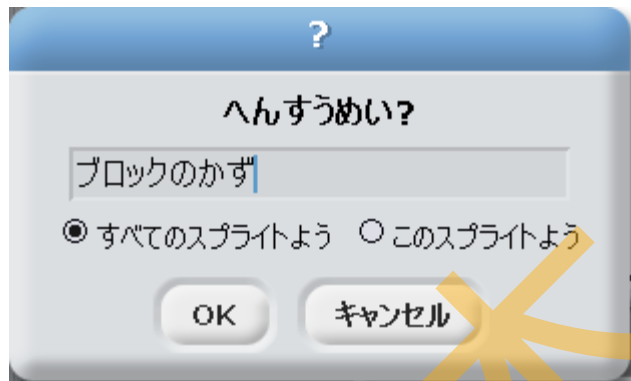
まだ、変数は作成していないはずですよ。

ここで「あたらしいへんすうをつくる」をクリックします。



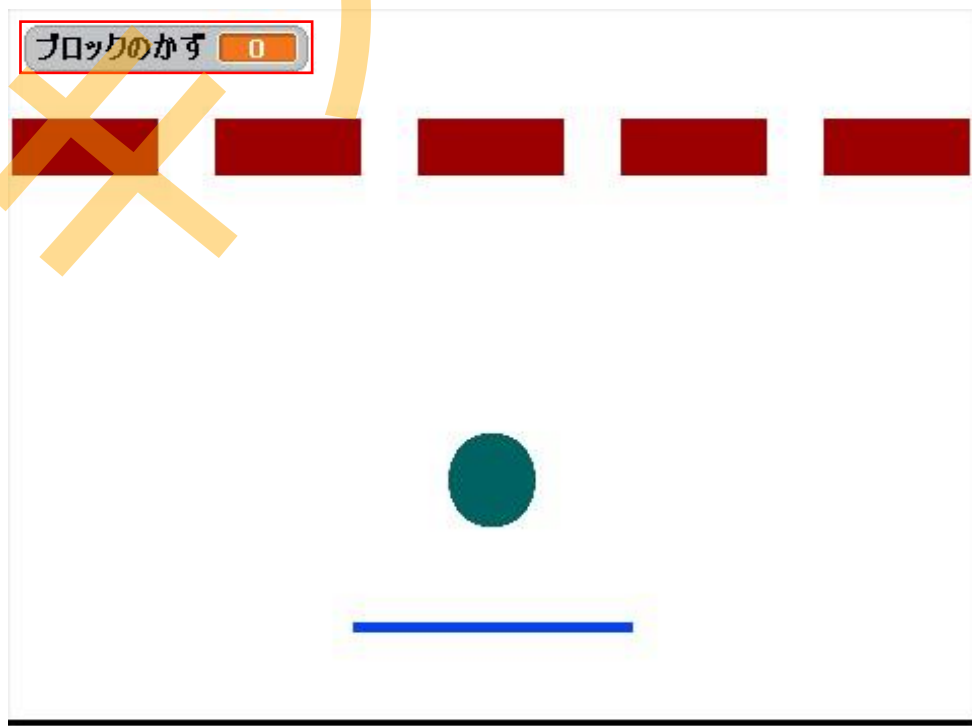
**2** 変数の名前を設定します。

名前は「ブロックのかず」にします。  
これで「OK」をクリックします。  
この名前はわかりやすいようにしておきましょう。



**3** ブロックパレットにパーツがいくつか追加されました。

また、ステージの左上にも「ブロックのかず」があります。これが数字を入れる箱になるのです。



次に「ブロックのかず」を 0 にする と 「ブロックのかず」を 1 ずつかえる について説明しましょう。

「ブロックのかず」を 0 にする

これは「ブロックのかず」という箱に自分で決めた数字を入れることができます。

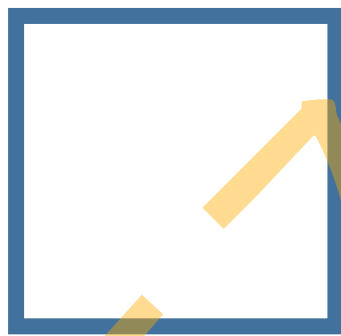
0の数字は変えることができます。

例えば、5にしてみましょ。

すると「ブロックのかず」という箱には5が入ります。

「ブロックのかず」を 5 にする

ブロックのかず



ブロックのかず



5を入れると……

もともと数字が入っている場合はその数字を取り出してから5を箱の中に入れます。

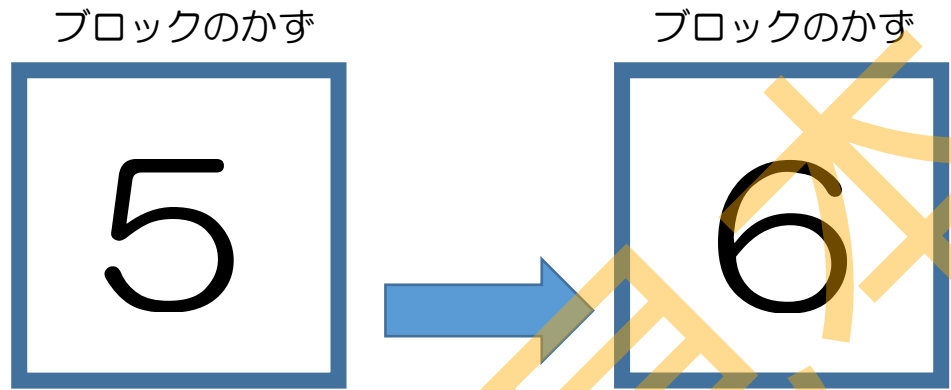
「ブロックのかず」を 1 ずつかえる

はもともと箱に入っている数字を加えて（減らして）、その数字を箱に入れ直します。

プラスの数字だと増えて、マイナスの数字だと減っていきます。



ブロックのかずを 1 ずつかえる



ブロックのかずを  
1 ずつかえるで

$$5+1$$

## 4. <sup>のこ</sup>残っているブロックをカウントしてみよう

それでは実際にこの変数を使ってブロックの数をカウントしてみましょう。  
まず、どのようなシチュエーションでブロックの数が変わるかは次のようになります。

### ● <sup>かいし</sup>ゲームを開始する時<sup>とき</sup>

ゲームを開始する時はブロックが5つです。

そのことをステージのプログラムにある

「ゲームスタートをおくる」の前に挿入します。

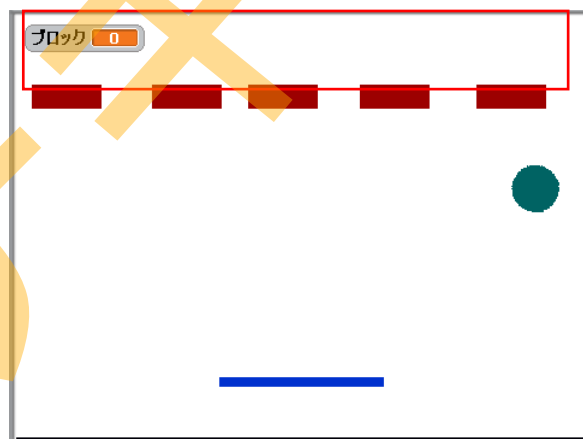
この5つのブロックがゲームを開始した時に登場  
します。

「ブロックのかず」という箱の中に5を入れましょう。



これを実行するためには「ブロックのかず」を0に設定し、

0を5に変えて挿入します。



ブロックのかず

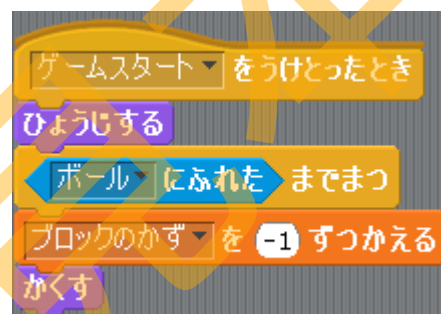
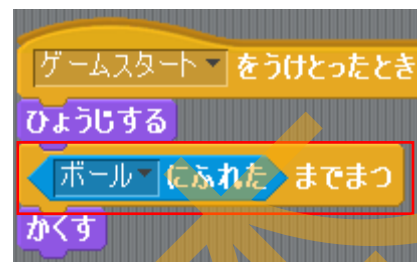


## ● フロックが消える時

ブロック 1~5 のプログラムを見てください。  
ブロックが消えるのはボールに当たった時です。  
ここでブロックが1つ減ることになります。  
つまり、「ブロックのかず」に入っている数字を1つ減らします。

これには「ブロックのかず」を1ずつかえるを使いましょう。

1つ減らすので「-1」にして挿入します。  
ブロック 1~5 も同じように変更します。



ブロックのかず



ブロックのかず



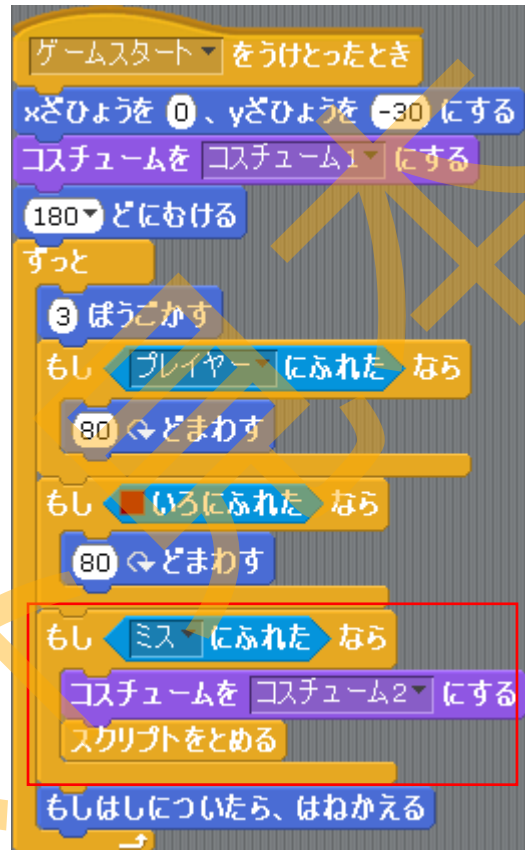
## 5. クリアした時にボールの動きを止めよう

ブロック崩しは下に落ちてミスした場合、ブロックを全て消してクリアした場合もゲームが終了するのでボールを止めなくてはなりません。

ミスした時のプログラムはすでにできていますが（右の図の赤枠）、クリアした場合もボールを止めてみましょう。

クリアした時はブロックが全てなくなった場合です。したがって、変数「ブロックのかす」が0になった時です。

どのように挿入していくかを説明しましょう。

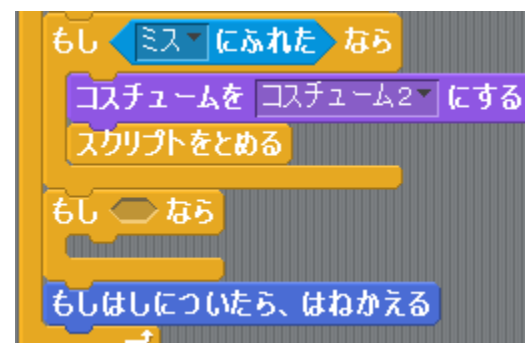


1 ボールのプログラムに右の図のように

もし  なら  
[ ] を挿入します。

この中の条件がポイントになります。

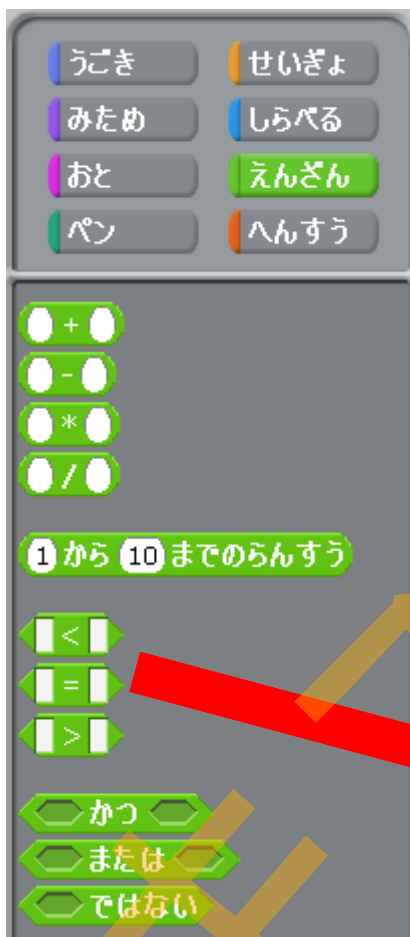
「ブロックのかす」が0を表すには「しらべる」ではなく、「えんざん」になります。



**2** 「えんざん」を見るときいろいろ命令のパーツがあります。

「ブロックのかず」が0ということは<ブロックのかず=0>ということです。

つまり、を使います。



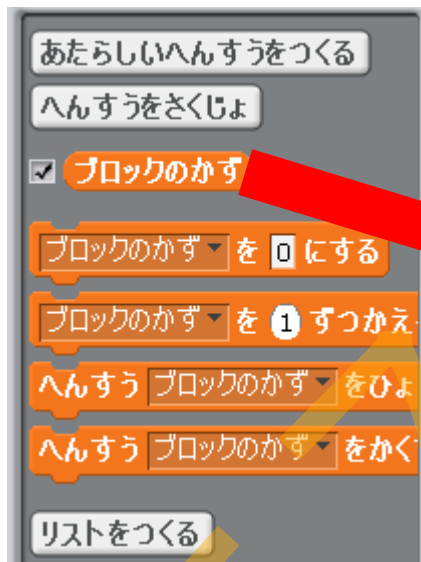
**3** 「ブロックのかず」はへんすうの中にある

**ブロックのかず** を挿入します。

**← = →** を使う場合は左右どちらに挿入しても

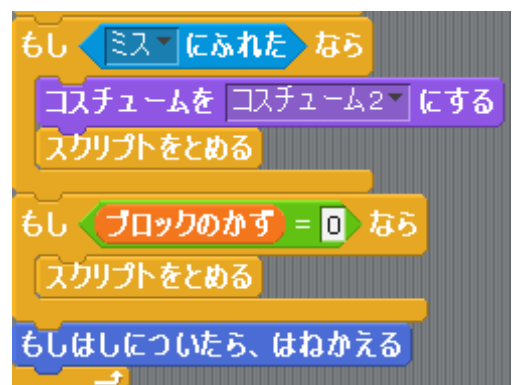
よいですがここは左に挿入します。

もう片方はキーボードから0を入力します。



**4** 「せいぎょ」にある「スクリプトをとめる」を挿入してプログラムを動かしてみましよう。

ミスをした時やブロックが全てなくなった時にボールは止まりましたか？



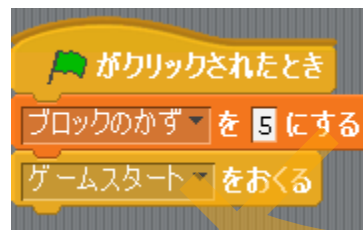
**?** ブロックの数を追加し

て 6個にした場合プログラムはどうすればいいでしょう？

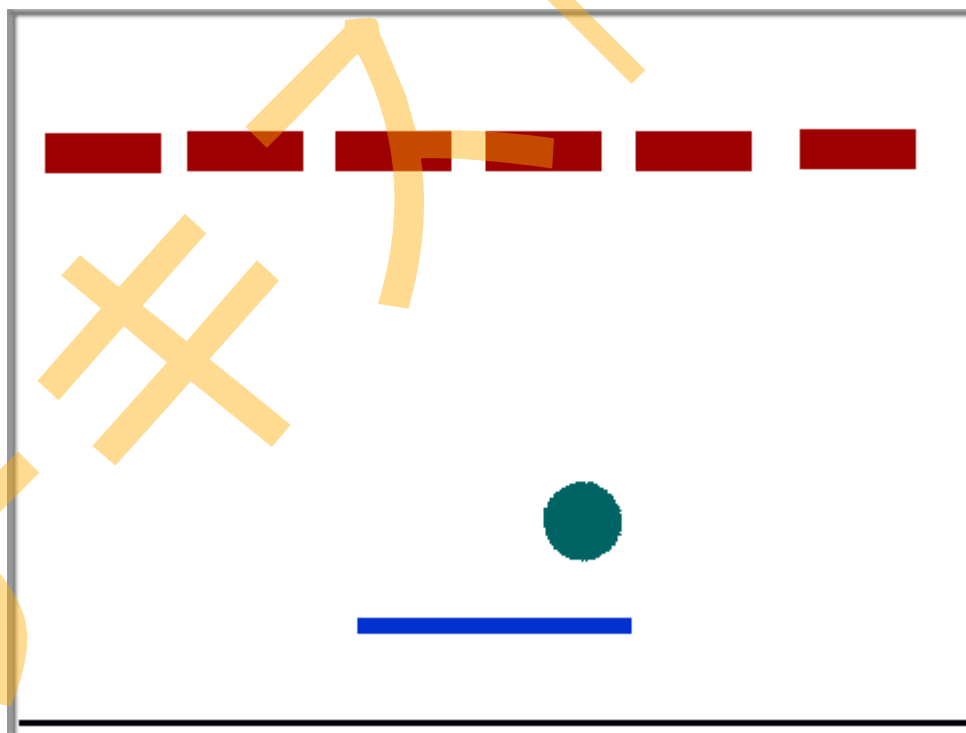
## ◇問題の答え◇

ブロックを6個に増やした場合、ゲームを開始した時にブロックの数が6になるのでステージのプログラムの「ブロックのかず」は6にします。

ブロックの数だけ、ゲーム開始の時の「ブロックのかず」を設定します。



6個目のブロックを下の図のように並べて実行してみましょう。



## 6. 音声を入れてみよう

次にいろいろなシチュエーションで音声を入れてみましょう。

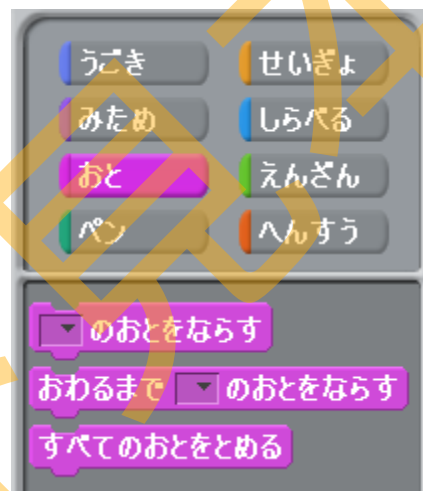
今回はボールをプレイヤーで打ち返した時、ボールが下に落ちてしまった時、クリアした時、ブロックが消える時に音を鳴らしてみよう。

音声を出すためには「**おと**」にある

「**のおとをならす**」を使います。

まずは音声を登録しないと音は出ません。

そのため、スプライトに対して登録を行います。



### ● 登録の手順

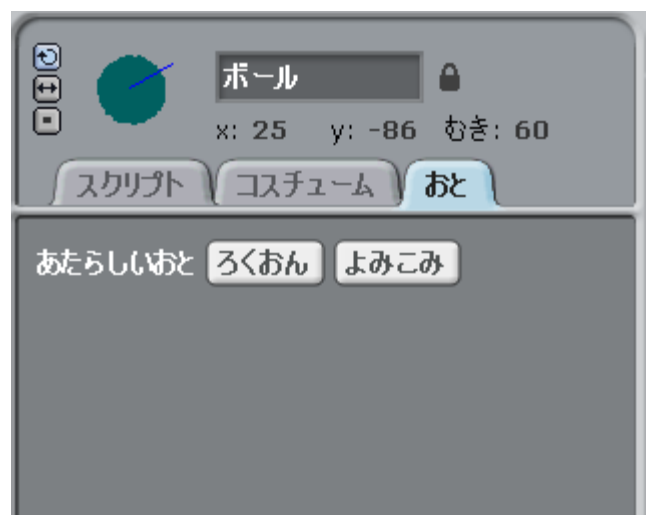
1 1 まず対象となるスプライト

(今回はボール)の「おと」をクリックします。

右の図はまだ登録されていません。

登録するにはあたらしいおとの「ろくおん」「よみこみ」をクリックします。

今回は「よみこみ」で登録してみよう。





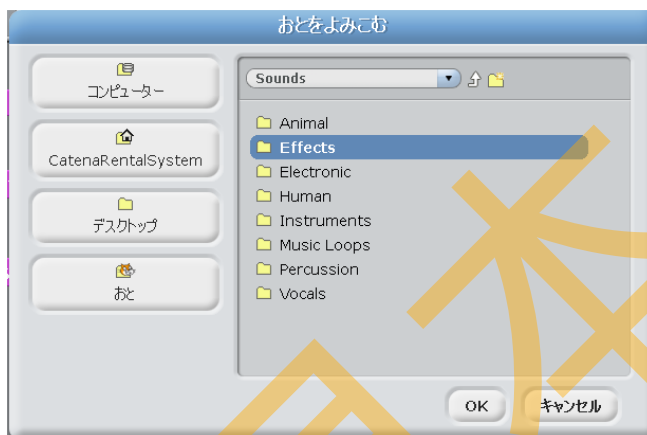
## 2 このようにおとをよみこむ

がめん で  
画面が出ます。

エフェクト  
Effectを開いてみましょう。

かく ひら おんせい  
各ファイルを開くと音声データフ

ァイルがひょうじ  
表示されます。



おんせい  
音声データファイルをクリックす  
るとおと かくにん  
音を確認できます。

えら おと ついか  
ファイルを選ぶとその音が追加さ  
れます。

ため ついか  
試しに1つ追加してみましょう。



## 3 エフェクト WaterDropを EffectsにあるWaterDropを 選択しました。

ばんめ ウォータードロップ とうろく  
1番目にWaterDropが登録されま  
した。

おんせい とうろく  
音声はいくつも登録することがで  
きます。

ををクリックすれば音が出ます。



また、右にある[X]をクリックすると削除することができます。

ボールのSpriteにはプレイヤーで  
打ち返した時、ボールが下に落ちてし  
まった時、クリアした時にそれぞれ違  
う音を出したいと思います。

したがって、3つの音声を用意します。  
右の図はあくまでも例です。

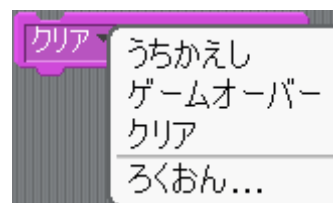
音声にも変数と同じように名前があり、  
変えることができます。

どこで使うか分かりやすい名前にして  
おくと便利です。

こんなふうにしておくと分かりやす  
いでしょう。



これで **のおとをならす** を挿入していきます。

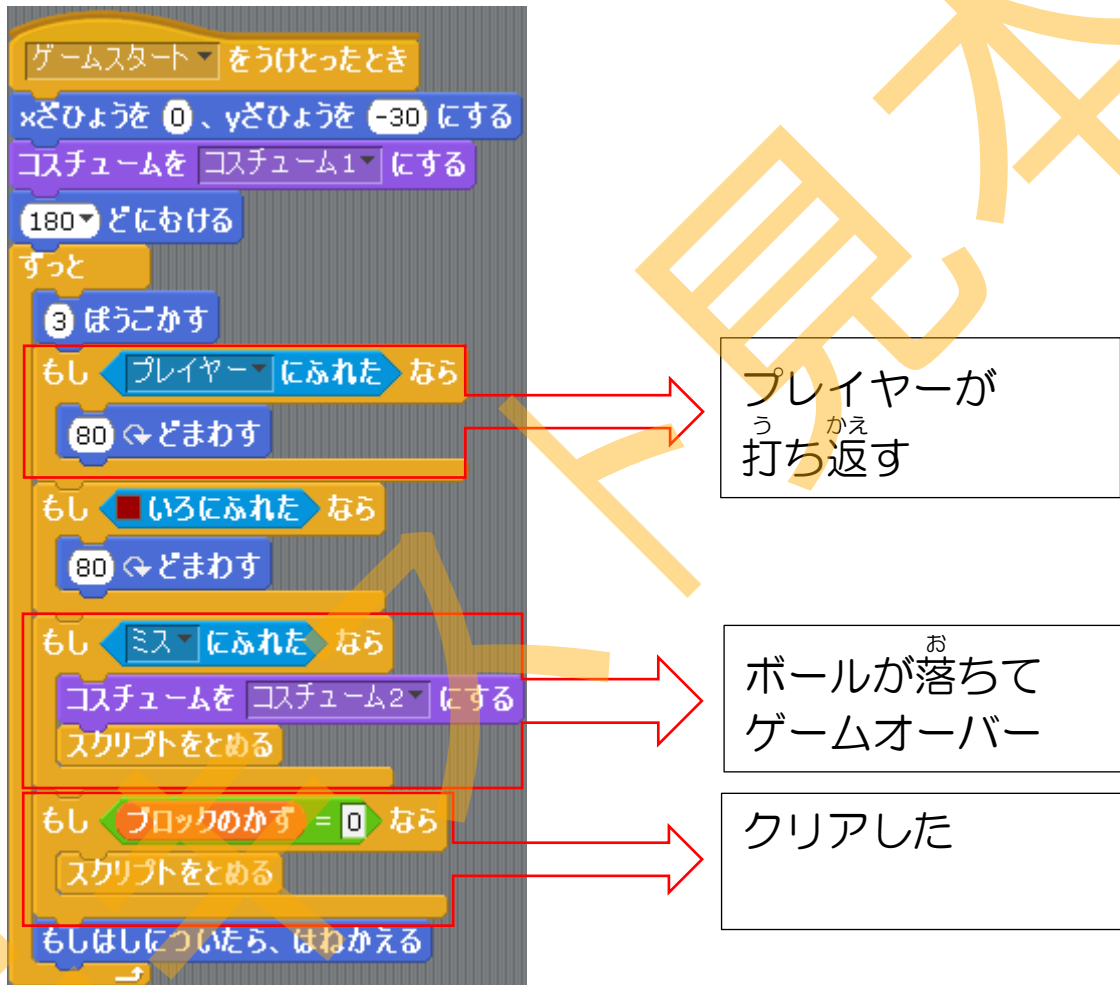


? この3つの音声をボールのプログラムのど  
こに挿入すればいいでしょう?

## ◇<sup>もんだい</sup>問題の<sup>こた</sup>答え◇

まずはプログラムの中で「もし……、ならば」がそれぞれどのような役割を持っているかをおさらいしましょう。

下の図のようになっています。



つまり、それぞれ挿入する場所は次のページのようにになります。

```

ゲームスタート をうけとったとき
xぞひょうを 0、yぞひょうを -30 にする
コスチュームを コスチューム1 にする
180 どのむける
ずっと
  ③ ぼうごかす
  もし プレイヤー にふれた なら
    80 どのまわす
  もし いろにふれた なら
    80 どのまわす
  もし ミス にふれた なら
    コスチュームを コスチューム2 にする
    スクリプトをとめる
  もし ブロックのかす = 0 なら
    スクリプトをとめる
  もしはしについたら、はねかえる

```

← うちかえし のおとをならす

← ゲームオーバー のおとをならす

← クリア のおとをならす

```

がクリックされたとき
xぞひょうを 0、yぞひょうを -30 にする
コスチュームを コスチューム1 にする
180 どのむける
ずっと
  ③ ぼうごかす
  もし プレイヤー にふれた なら
    80 どのまわす
    うちかえし のおとをならす
  もし いろにふれた なら
    80 どのまわす
  もし ミス にふれた なら
    コスチュームを コスチューム2 にする
    ゲームオーバー のおとをならす
    スクリプトをとめる
  もし ブロックのかす = 0 なら
    クリア のおとをならす
    スクリプトをとめる
  もしはしについたら、はねかえる

```

次にブロックを消える時に音を出してみましょう。

これはブロックのプログラムで行いますので、ブロックのSpriteの「おと」に音を登録します。

例では右の図「よみこみ」から「Percussion」にある「HandClap」を読み込みます。



ボールのプログラムと同じように挿入しましょう。

他のブロックにも同じように挿入して動作を確認してみましょう。

音は出ましたか？



? ブロックの消える音や打ち返しの音などを変更するにはどうしたらいいでしょう？

## ◇問題の答え◇

別の音に変えたい場合は新しく音声のファイルを読み込みます。

例として、打ち返しの音を変えてみましょう。

まずはボールのSpriteの「おと」

に新しい音声を登録します。

例としてElectronicにあるZoopという

音声を登録します。



名前を「うちかえし」にしたいのですが音声中に同じ名前をつけることはできません。

したがって、もともとある「うちかえし」は名前を変えるか、削除しなくてはなりません。

例では[X]をクリックして削除します。



「Zoop」を「うちかえし」に変えましよう。



これで実際にプログラムを動かして音声がかわっているか確認しましよう。他にもいろいろな音を鳴らしてみましよう。



# 1. ステージを作ろう

中級編でここまではブロック崩しのスタートからクリア、ミスの流れまで作りました。ここからはブロック崩しを面白くするためにステージを切り替えましょう。ステージを作る上でしていきたいことは以下のようにになります。

- ステージごとに背景を変える
- ステージごとに設置するブロックの数を考える
- ステージごとにボールの速さを変える

まずはステージを作るために基本的なプログラムを作成します。

## ● なんこめ 何個目のステージかを表すには変数を使う

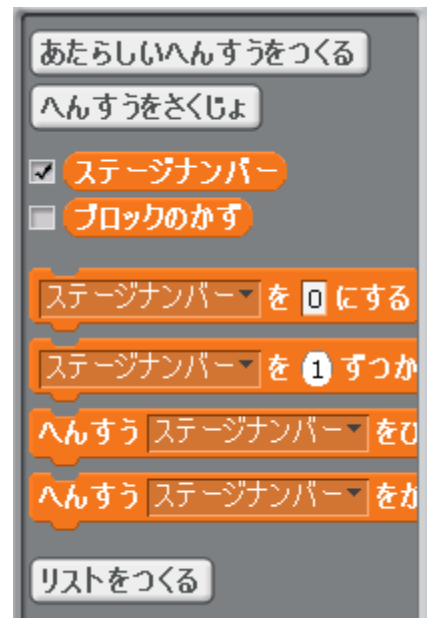
変数は数字を入れておく箱であることを前に説明しました。

その変数がここでも登場します。

ゲームでは「ステージ1、ステージ2……」という言い方があると思います。

そのステージ数をカウントするために変数を1つ用意しておきます。

「ステージナンバー」という変数を用意します。

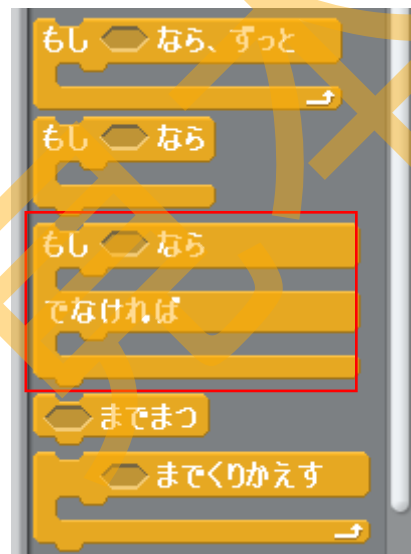




# ● ステージの切り替えはステージ（背景）で 作成する

はいけいの切り替えはステージのスク립トで行います。

- 1** まずはステージを2つ作ることにしましょう。  
ステージごとに分けると考えます。  
つまり、もし「ステージナンバー」が1であれば、  
もし「ステージナンバー」が2であればという考え  
方です。  
もしもの場合が2つ以上ある場合は「もし……であ  
れば、でなければ」を使います。

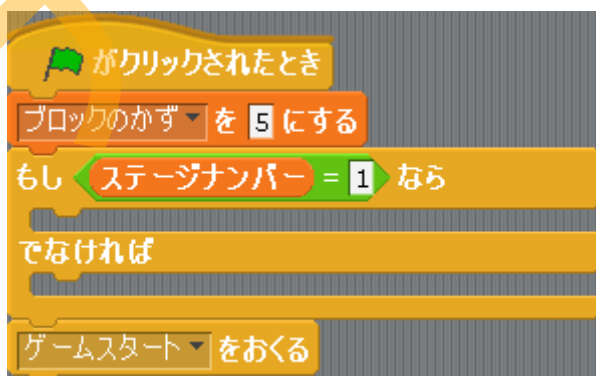


- 2** 条件にまず、「ステージナンバー」が  
1を挿入します。  
これで「ステージナンバー」が2の場合は  
「でなければ」の命令で動くようになります。



ステージナンバーが2の時

これをステージのプログラムの「ゲームスタートをおくる」の前に挿入しましょう。



「ステージナンバー」が1の時

「ステージナンバー」が2の時

**3** 「ステージナンバー」の数字を「もし……であれば、でなければ」の前に設定しておく必要があります。

したがって、「ステージナンバーを0にする」を挿入します。0になっている数字は1にしておきましょう。



プログラムを動かす時はこの命令で「ステージナンバー」の数字を変えて行います。

## 8. ステージの背景を変えてみよう

まずはステージの数だけ背景を用意しなくてはなりません。

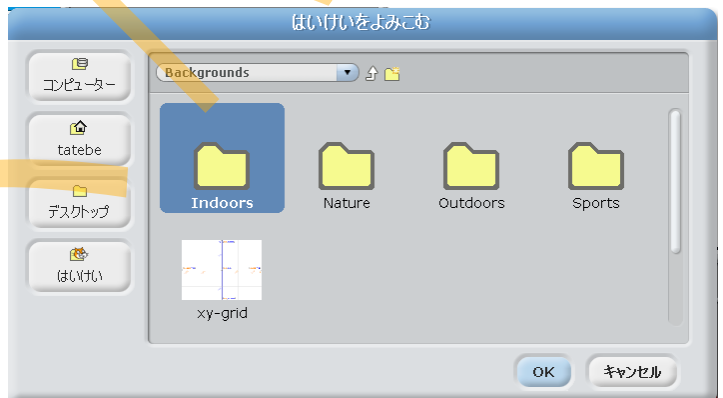
ステージの「はいけい」をクリックすると真っ白な背景が1つだけあります。

「あたらしいはいけい」の「よみこみ」から背景を読み込みます。

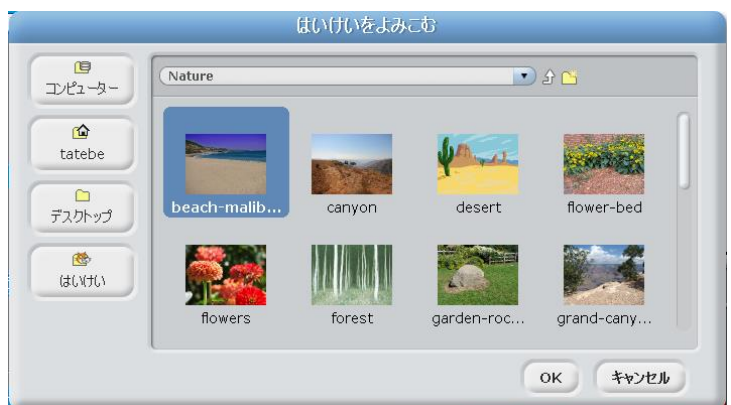


音声の時と同じようにファイルを  
読み込む画面が表示されます。

画像ファイルは画像が表示されま  
す。



例えばNatureを開くと右の図のよ  
うに表示されます。



2つのステージを作成しますから  
背景を2つ選びます。

今回はOutdoorsにあるboardwalkと  
Natureにあるunderwaterを選びます。



音声と同じように背景にも名前がつ  
けられます。

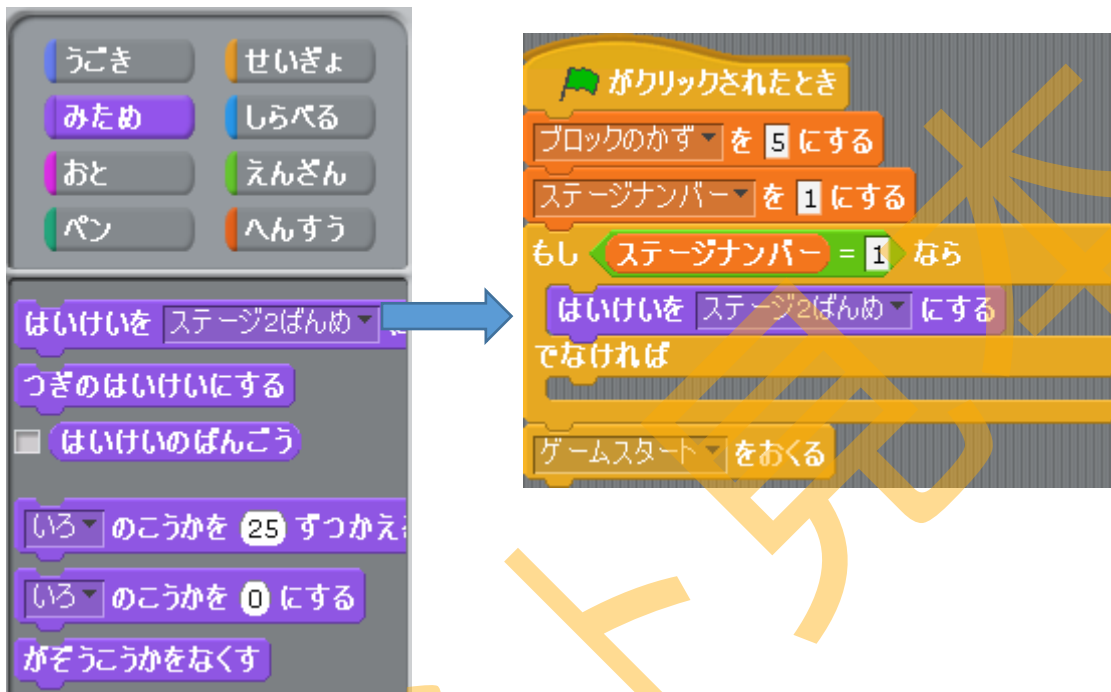
右の図のように名前をつけるとわか  
りやすいでしょう。

背景が用意できたらステージごとに  
背景を設定する手順を説明しましよ  
う。



**1** 「みため」から「はいけいを……にする」を挿入します。

まずはくもしステージナンバー=1なら>に挿入します。



挿入ができれば「ステージ 1ばんめ」に変えておきましょう。



**2** 次に「<でなければ>」に「はいけいを……にする」を挿入します。「ステージ 2 ばんめ」にしておきましょう。



これでプログラムを動かしてみましよう。なお、プログラムを動かすには「ステージナンバー」を 0 にするで数字を設定しましよう。

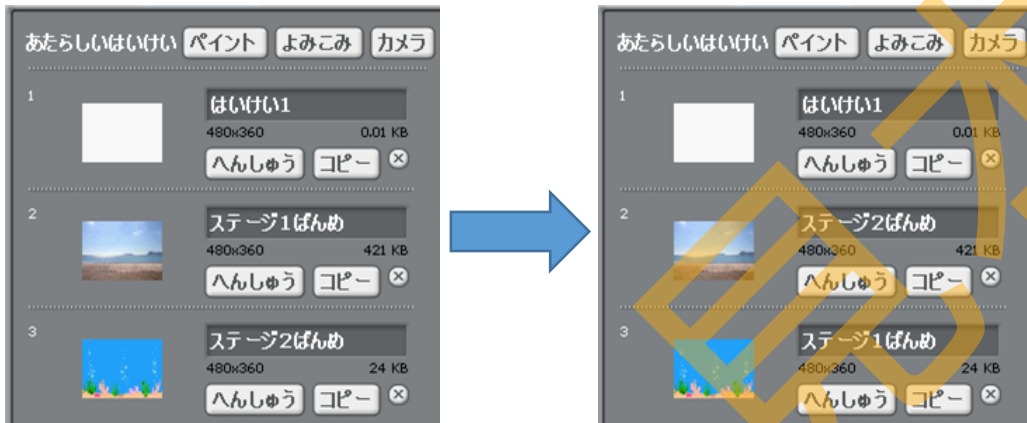
指定した背景になるかを確認しましよう。

**?** もしステージ 1 の背景とステージ 2 の背景を入れ替えたい場合はどうすればいいでしょう？

## ◇問題の答え◇

いろいろな方法があります。

1つ目の方法は「はいけい」において登録している背景の名前を入れ替えることです。

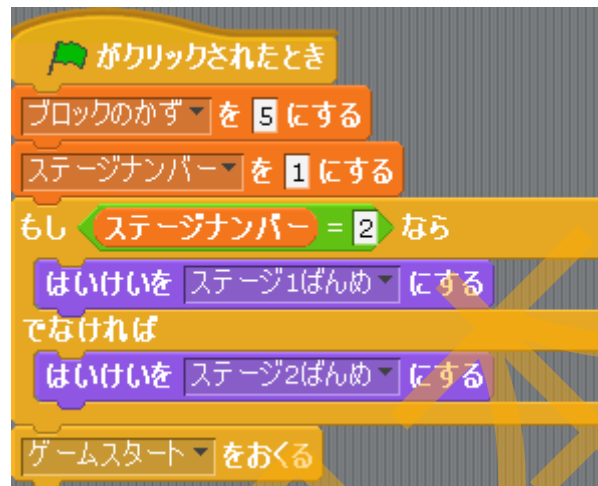


2つ目の方法はプログラムを変えます。プログラムは2通りの考え方があります。プログラムではステージナンバーが1の時に背景を「ステージ1ばんめ」、そうでなければ「ステージ2ばんめ」にしていますから入れ替えるにはこのように考えます。

ステージナンバーが1の時に背景を「ステージ2ばんめ」、そうでなければ「ステージ1ばんめ」



ステージナンバーが 2 の時に背景  
を「ステージ 1 ばんめ」、そうでな  
ければ「ステージ 2 ばんめ」を挿入  
します。





## 9. ステージごとにブロックの数を増やそう

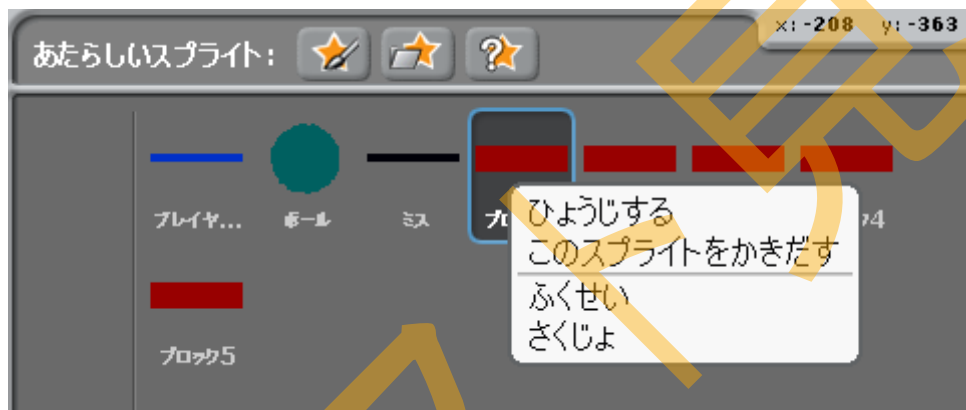
ゲームではステージが進むほど難しくしなくてはなりません。

そこでステージが進むたびにブロックを増やしてみましょう。

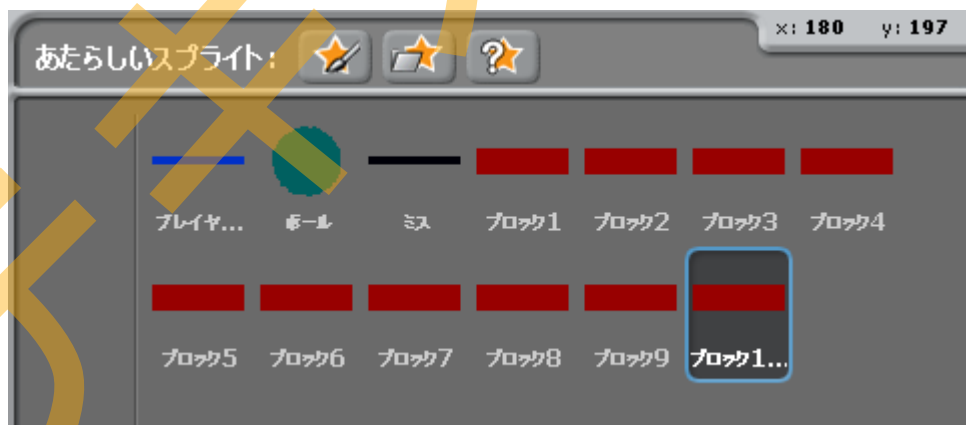
ステージ 1 では 5、ステージ 2 では 10 にしてみましょう。

まずは同じブロックを 10 まで増やしてみましょう。

スプライトを右クリックして「ふくせい」を選べば、スプライトをコピーすることができます。



名前はブロック 6、ブロック 7、ブロック 8……、というふうにしていきます。



ブロックを10にしたら下の図のように並べましょう。



並べ方は上は左からブロック1、ブロック2、ブロック3、ブロック4、ブロック5、下は左からブロック6、ブロック7、ブロック8、ブロック9、ブロック10

ブロック6からブロック10の座標を以下のようにしましょう。

ブロックの名前	X座標	Y座標
ブロック6	-200	80
ブロック7	-100	80
ブロック8	0	80
ブロック9	100	80
ブロック10	200	80

下のブロックはステージ2から登場するようにします。そのためにはステージのプログラムとブロックのプログラムに命令を挿入していきます。

## ● ステージのプログラム

- 1 「へんすう」から「ブロックのかずを0にする」を「はいけいを……にする」と同じところに挿入おなします。

あたらしいへんすうをつくる  
へんすうをさくじよ  
 ステージナンバー  
 ブロックのかず  
ブロックのかず を 0 にする  
ステージナンバー を 1 ずつか  
へんすう ステージナンバー をひ  
へんすう ステージナンバー をか  
リストをつくる

がクリックされたとき  
ブロックのかず を 5 にする  
ステージナンバー を 1 にする  
もし ステージナンバー = 1 なら  
はいけいを ステージ1ばんめ にする  
ブロックのかず を 0 にする  
でなければ  
はいけいを ステージ2ばんめ にする  
ブロックのかず を 0 にする  
ゲームスタート をおくる

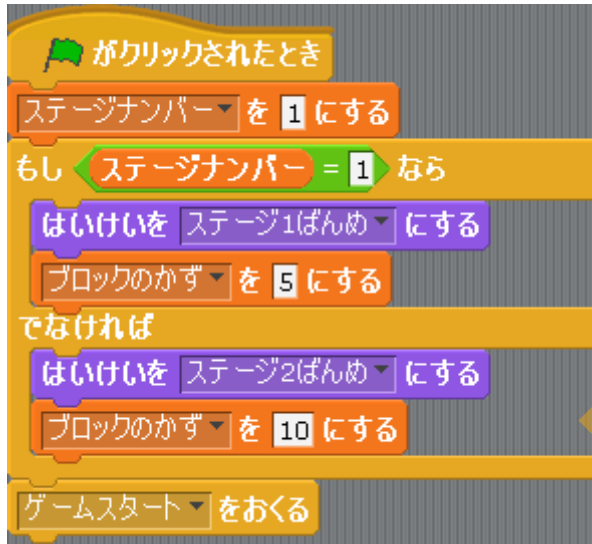
ステージナンバー を 0 にする になっている  
場合は「ブロックのかず」にか変えましょう。

ステージナンバー	ステージナンバー ブロックのかず
----------	---------------------

- 2 ステージ 1 では 5、ステージ 2 では 10 にします。

がクリックされたとき  
ブロックのかず を 5 にする  
ステージナンバー を 1 にする  
もし ステージナンバー = 1 なら  
はいけいを ステージ1ばんめ にする  
ブロックのかず を 5 にする  
でなければ  
はいけいを ステージ2ばんめ にする  
ブロックのかず を 10 にする  
ゲームスタート をおくる

3  がクリックされたとき の次にある  ブロックのかず を 5 にする は削除しておきましょう。



## ● フロックのプログラム

した  
下のブロックのプログラムにも  
あら  
新たな命令を挿入しなくてはな  
りません。

した  
下のブロックはステージ 2 にの  
み登場します。

したがってステージ 2 の時だけ  
ブロックが表示されるように  
命令を出さなければなりません。

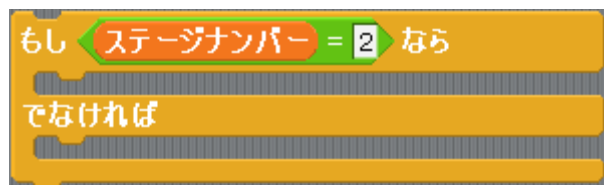
めい  
命令を出すのは下のブロックである「ブロック 6」、「ブロック 7」、「ブロック 8」、「ブロック 9」、「ブロック 10」になります。

その方法を説明していきましょう。



**1** ブロックが登場するように命令して  
いるのはブロックのスクリプトにある  
「ひょうじする」です。ここを変えます。  
まず、「せいぎょ」「もし……なら、でな  
ければ」をブロック 6 のスクリプトに置きま  
す。

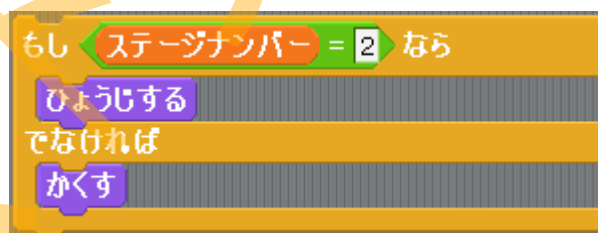
条件は「ステージナンバー=2」になります。



**2** 「ゲームスタートをうけとったとき」の下にある「ひょうじする」をとって「もしステージナンバー=2なら」に挿入します。



**3** 「でなければ」に「みため」の「かくす」を挿入します。



**4** 「ゲームスタートをうけとったとき」の後に「もしステージナンバー=2なら」をつなぎ、その下に「ボールにふれたまでまつ」をつなぎます。これをブロック7からブロック10のスプリクトにも行います。

ステージナンバーを1にした場合と2にした場合にブロックの数が違うかを確認してみましょう。



## 10. ステージごとにボールの<sup>はや</sup>速<sup>か</sup>さを<sup>か</sup>えよう

ゲームを<sup>むずか</sup>しくするために<sup>つぎ</sup>次の<sup>すす</sup>ステージに進んだ<sup>とき</sup>時にボールの<sup>か</sup>スピードを変えて<sup>みま</sup>みましょう。

まず、ボールの<sup>か</sup>スピードはプログラムの<sup>どこ</sup>どこで<sup>ま</sup>設定しているか<sup>を</sup>を見て<sup>みま</sup>みましょう。

実際に<sup>じつ</sup>ボールが<sup>うご</sup>動くのは「3 <sup>ぼう</sup>ぽうご<sup>か</sup>す」があるから<sup>です</sup>です。

この「3」は<sup>じゆう</sup>自由に<sup>にゆうりよく</sup>入力<sup>を</sup>することができます。実は<sup>じつ</sup>この<sup>なか</sup>中には<sup>へんすう</sup>変数<sup>を</sup>入れる<sup>こ</sup>ことができます。

<sup>へんすう</sup>変数は<sup>じゆう</sup>自由に<sup>にゆうりよく</sup>数字<sup>を</sup>入れる<sup>こ</sup>ことができますので<sup>へんすう</sup>変数<sup>を</sup>使<sup>つ</sup>って<sup>か</sup>ボールの<sup>か</sup>スピード<sup>を</sup>変えて<sup>みま</sup>みましょう。

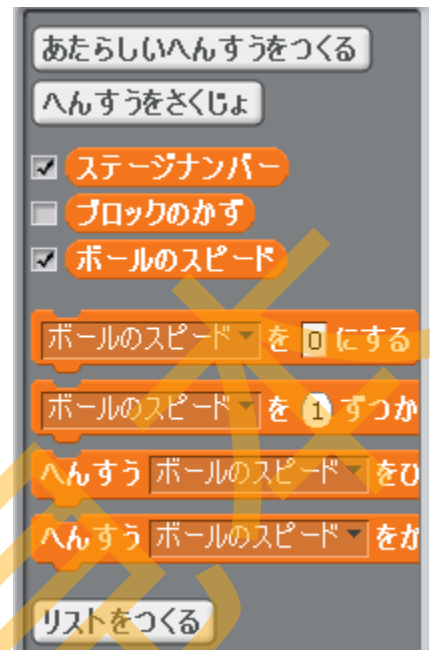


まずは変数を用意しましょう。

「ボールのスピード」という変数を用意しました。

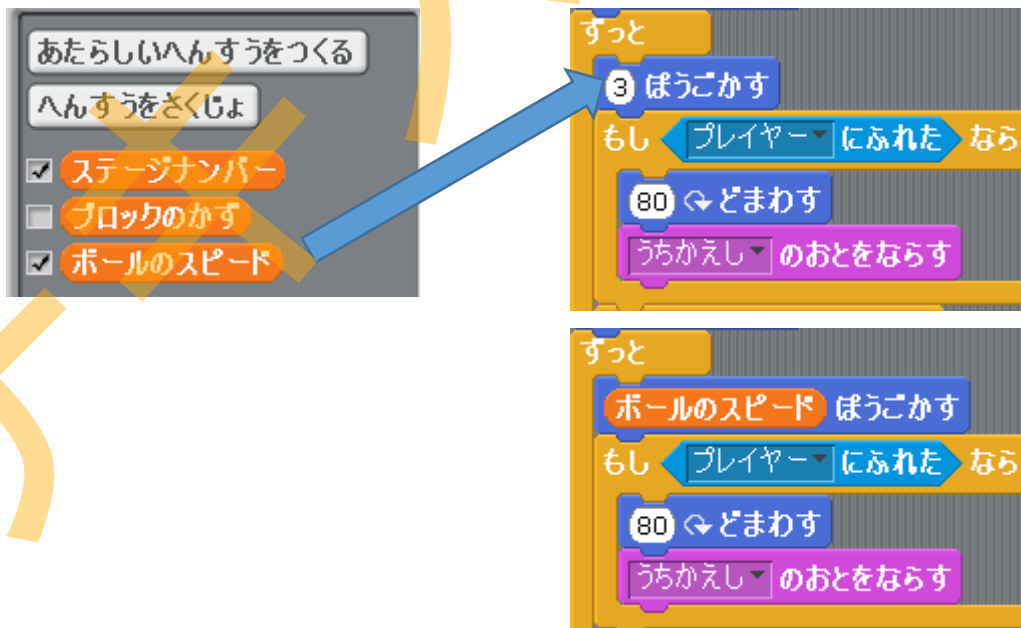
この「ボールのスピード」を使って実際に速さを変えてみましょう。

まずはスピードを変えてみるということに挑戦してみましょう。



**1** 実際にボールのプログラムの「3 ぼうごかす」に変数を挿入してみましょう。

「へんすう」にある「ボールのスピード」をクリックしてドラッグし、「3 ぼうごかす」の場所で放せばいいだけです。





**2** プログラムを動かします。ボールが動いている時にブロックエリアにある

**ボールのスピード** を **0** にする の数字を変えてクリックして変数の数字を変えてみましょう。数字を大きくしたり、小さくしたりするとボールの速さがどのように変わるかを確かめてみましょう。

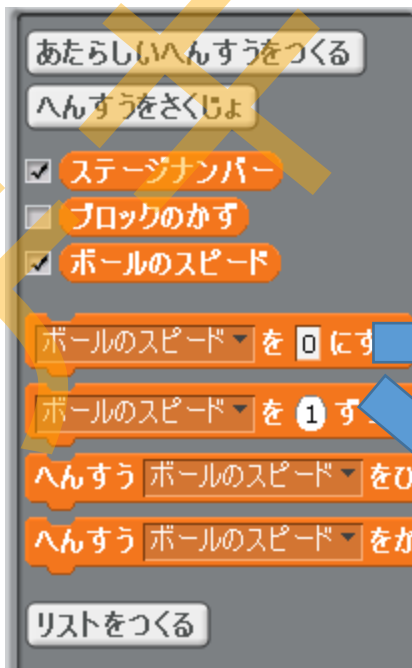
変数の数字を変えることでボールの速さが変わることが確認できれば次にステージごとに速さを設定してみましょう。

ステージごとにボールの速さを設定する場合は背景やブロックの数と同じように行います。

したがって、ステージのプログラムで<もし……ならば、でなければ>に挿入していきます。



**3** 「ボールのスピードを0にする」を「もし……ならば、でなければ……」に挿入します。



**4** 「ボールのスピード」の変数の数字  
を変えましょう。

数字が大きければ大きいほどボールは  
速くなります。

したがって、ステージナンバーが2の時  
は1の時よりも大きくしましょう。




これでプログラムを動かしてステージナンバーが1の時と2の時はボールのスピ  
ードが違うか確かめてみましょう。

# 11. 障害物を作ってみよう

ステージ 2 からは障害物を登場させてみましょう。今回作る障害物は一定時間に左から右に移動して、ボールに当たるとボールの進行方向が変わるというものです。

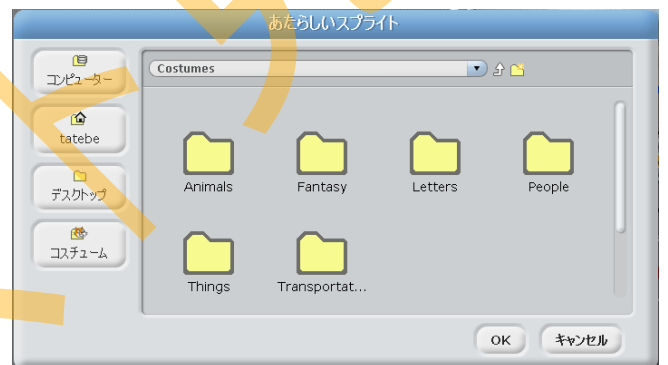
まずは sprites を 1 つ用意します。

ここでは sprites をファイルから読み込みます。

あたらしい sprites の  をクリックします。



ファイル選択画面が表示されます。試しに Animals を開いてみましょう。



画像ファイルは画像で表示されます。



今回は Fantasy から ghost2-a を選んでみました。



このようにスプライトがステージに登場しますがボールに比べると明らかに大きすぎます。これはプログラムの中で小さくしましょう。

それではまず、小さくしてから左から右にいくプログラムを作成しましょう。

スプライトの名前は「しょうがいぶつ」にしましょう。



まずはボールのプログラムで障害物に当たった時の命令を挿入していきます。

**1** 「せいぎょ」から「もし……なら」を「ずっと」の中にもう1つ追加しましょう。

右の図のように挿入しましょう。

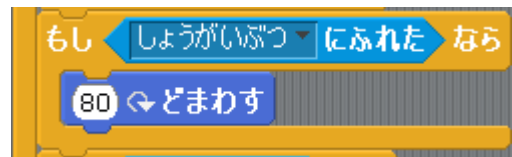


**2** 条件は「しょうがいぶつにふれた」です。「しらべる」から「……にふれた」を挿入して、「しょうがいぶつ」にします。



**3** 角度を変える命令を挿入します。

ブロックやプレイヤーに当たった時と同じように「80 どまわす」を挿入します。



**4** 音声を入れましょう。

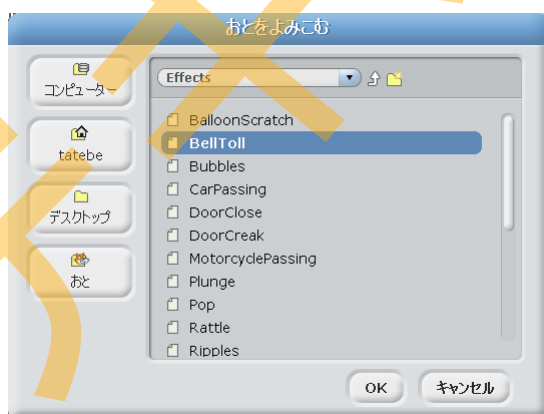
ブロック、プレイヤーが当たった時の音とは違うものにしましょう。

そのために「おと」に移動します。

「あたらしいおと」の「よみこみ」をクリックしてファイル選択画面を出してファイルを選びましょう。



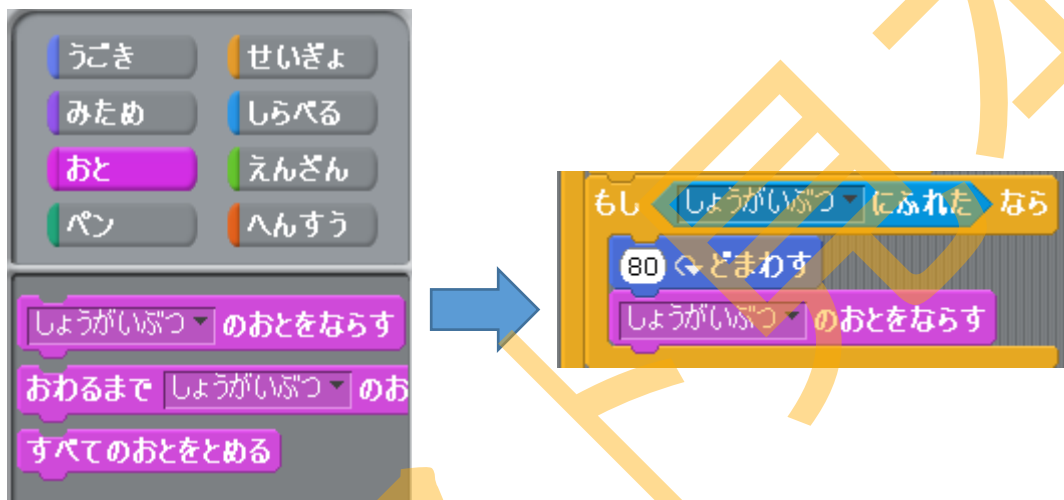
今回はEffectsもあるBellTollを選択しました。



わかりやすいように名前は「しょうがいぶつ」  
にしておきましょう。



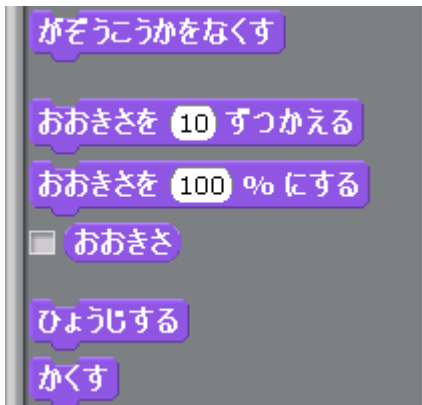
スクリプトに戻ってこの音声が鳴る命令を挿入します。「おと」にある「しょうがいぶつのおとをならす」を挿入しましょう。



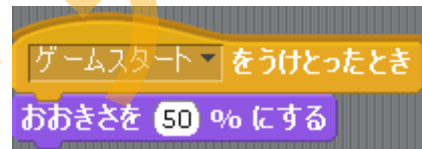
これで実際に障害物を置いてボールが当たった時にどのように動くのか確かめてみましょう。

1 障害物もボール、プレイヤー、ブロックと同じようにゲームスタートというメッセージを受け取ってから動くようにします。

大きさは「みため」にある「おおきさを 100%にする」で変更することができます。これを挿入しましょう。



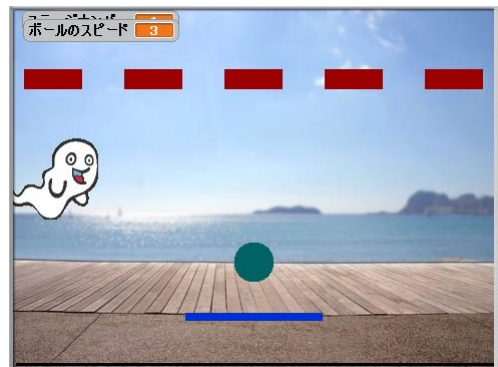
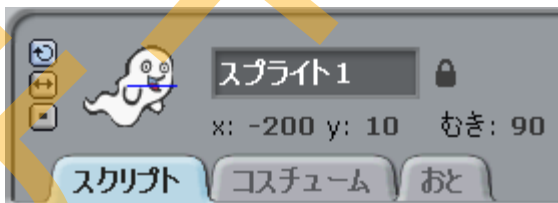
大きさは 50%ぐらいにしておくといいでしょう。



次に大きさを 50%にしたスプライトを左端や右端に寄せて、左端にいる時の x の座標と右端にいる時の x の座標を求めます。

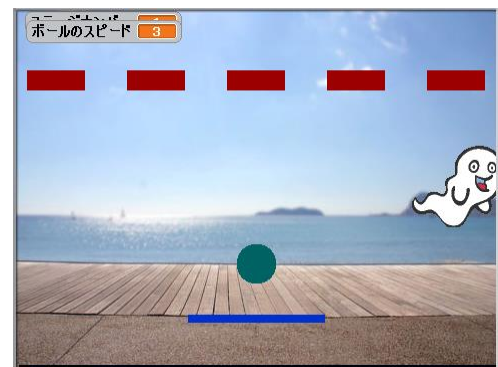
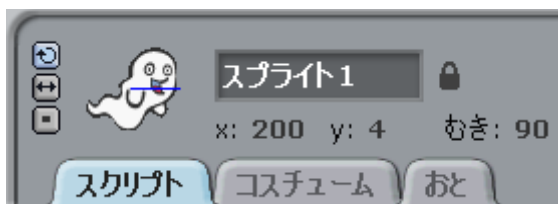
左端の x の座標は -200 にします。

これが出現する位置になります。



右端の x の座標は 200 にします。

これが姿を消す位置になります。

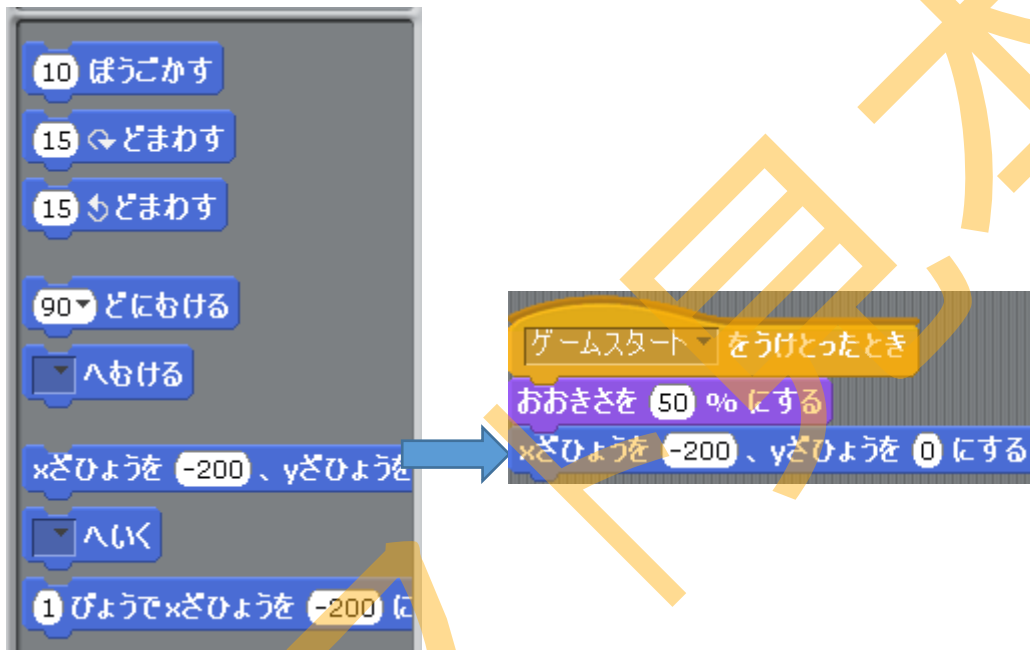


## 2 障害物を左端で出現させます。

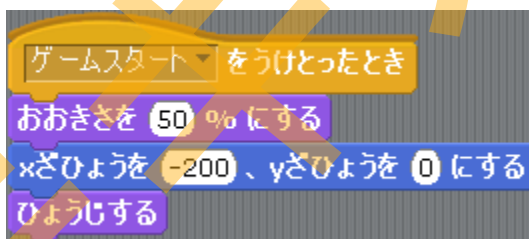
障害物を出現させる前に先に位置を設定することが大事になります。

位置は「うごき」の「xざひょうを……にする、yざひょうを……にする」です。

これを挿入してx座標は左端の-200、y座標は真ん中の0にしておきましょう。



次に「みため」から「ひょうじする」を挿入します。



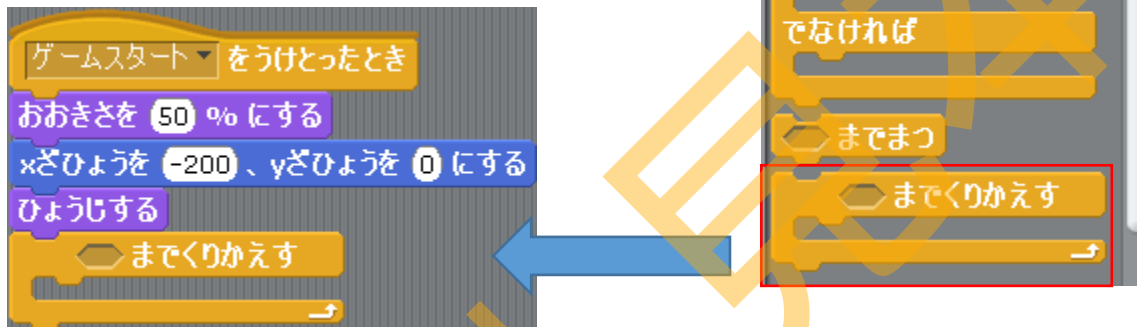


1 次に右端まで移動するという命令を出さなくてはなりません。

そのために「うごく」ということを右端に行くまで繰り返します。このような繰り返しに使うのが「せいぎょ」の「までくりかえす」です。

これは条件を満たすまで繰り返すというものです。

これを挿入しましょう。



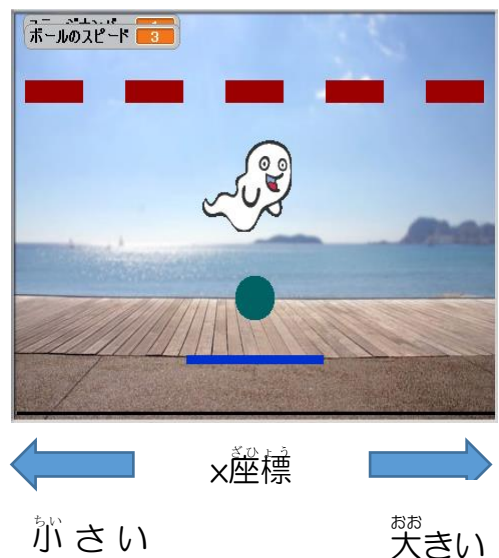
2 条件は「右端に着く」までです。






これをどのように表現するかは先程、求めた右端のx座標になります。

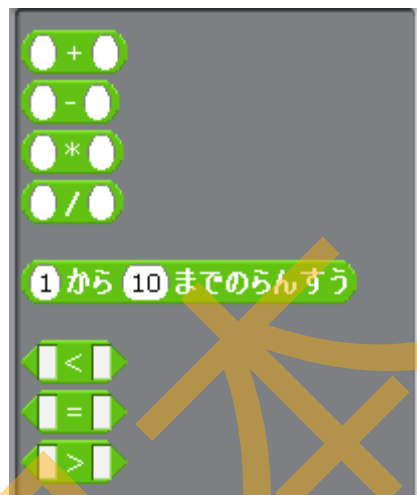
x座標は右になればなるほど数字が大きくなります。したがって、右に進むということはx座標の数字が大きくなるということになります。右端に着くということは右端のx座標よりも大きくなった場合です。

つまり、ここに条件は「スプライトのx座標が右端のx座標より大きくなるまで」繰り返せばいいのです。

これを表現するためには、まず「えんざん」を見てみましょう。



「えんざん」を見ると  と  があります。  
 は右の方が大きいという意味になっています。  
 は左の方が大きいという意味になっています。  
 今回は  を使しましょう。

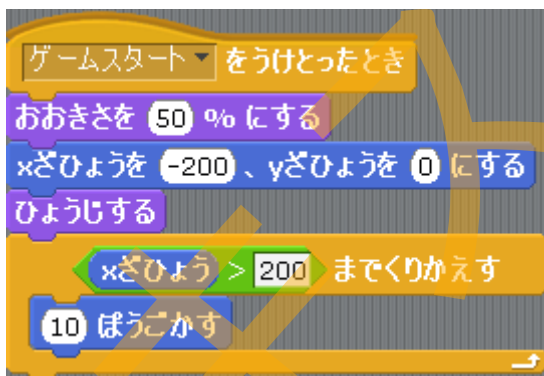
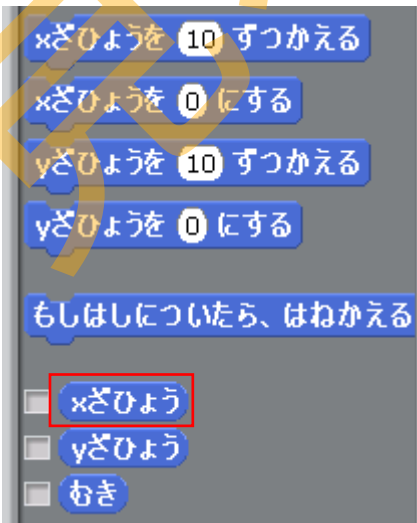


「Spriteの x座標が右端の x座標より大きくなるまで」ですから左側は「うごき」にある「xざひょう」になります。

右側は右端の x座標になりますから 200 になります。

したがって  になります。

これを条件に挿入しましょう。



**3** 障害物を動かす命令を挿入しましょう。

「うごき」から「ほうごかす」を挿入します。

数字は5くらいにしておきましょう。

ゆっくり動かせるために「せいぎょ」から「びょうまつ」の命令を挿入します。

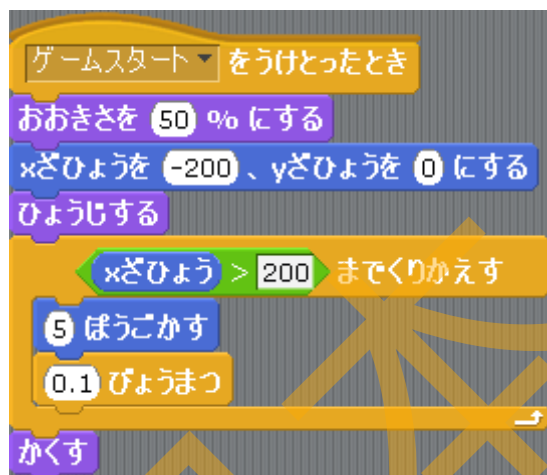
秒数は0.1にしておきましょう。



**4** 最後に障害物をステージから消してみましょ。

「みため」から「かくす」を繰り返しの後に挿入します。

これでプログラムを動かして、障害物が動くか、ボールに当たるとボールが角度を変えるか確かめてみましょう。



さて、このプログラムではステージナンバーを1にした場合でもゲームが始まるとすぐに障害物が登場します。

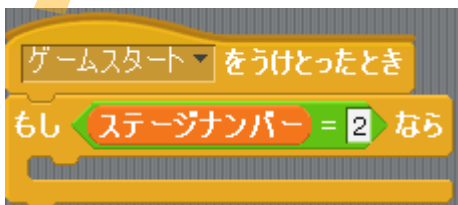
そして、一度右端に着いて姿を消してしまうとそれ以降障害物は現れなくなってしまう。

そこでステージナンバーが2の場合のみ、登場するようにして、一定時間ごとに出現するようにしたいと思います。

**1** まずは<ステージナンバー=2>の時にこの障害物が動くようにしましょう。

まずは「ゲームスタートをうけとったとき」と「おおきさ50%にする」の間を切り離します。そして、「せいぎょ」から「もし……なら」を「ゲームスタートをうけとったとき」の後に挿入します。

条件は<ステージナンバー=2>です。



**2** 「せいぎょ」から「ずっと」をスクリプト内に入れて「おおきさ 50%にする」  
る」



The script starts with a 'ずっと' (forever) loop block. Inside the loop, the following blocks are stacked: 'おおきさを 50%にする' (set size to 50%), 'xざひょうを -200、yざひょうを 0にする' (set x and y coordinates), 'ひょうじする' (show), a 'xざひょう > 200' condition with 'までくりかえす' (repeat until) block, '5 ぼうごかす' (shoot), '0.1 びょうまつ' (wait), and 'かくす' (hide).

いこう  
以降を「ずっと」の中  
なか  
に挿入します  
そうにゆう



The script is identical to the one above, showing the 'ずっと' loop with its internal blocks.

**3** 「ずっと」を「もし……なら」に挿入し  
ます。



The script starts with a 'ゲームスタート' (when game starts) trigger block, followed by a 'もし ステージナンバー = 2 なら' (if stage number is 2) condition. A blue arrow points from the 'もし' block to the 'ずっと' loop block. The 'ずっと' loop contains the same sequence of blocks as in step 2: 'おおきさを 50%にする', 'xざひょうを -200、yざひょうを 0にする', 'ひょうじする', 'xざひょう > 200' condition with 'までくりかえす', '5 ぼうごかす', '0.1 びょうまつ', and 'かくす'.

右の図のようになります。

これでプログラムを実行すると右端に着いてもすぐに左端から障害物が出現します。

これを一定の秒数を開けてから出現するようにしましょう。

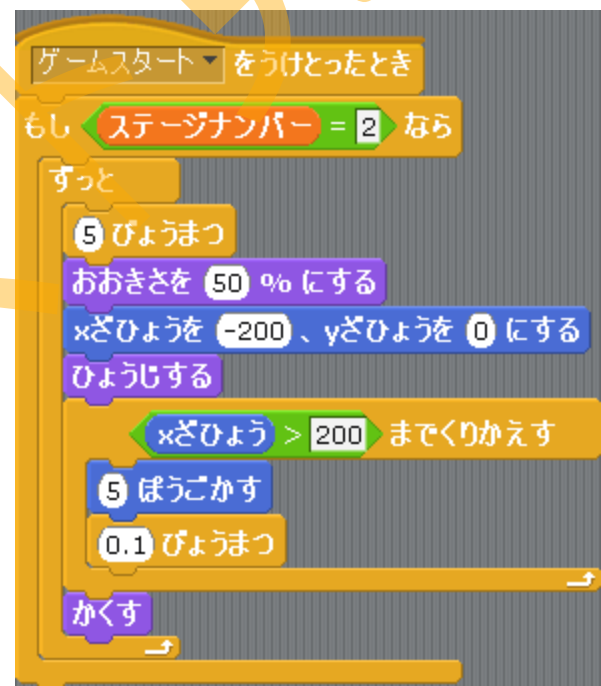


```
ゲームスタート ▾ をうけとったとき
もし ステージナンバー = 2 なら
  ずっと
    おおきさを 50 % にする
    xざひょうを -200、yざひょうを 0 にする
    ひょうじする
    <xざひょう > 200 までくりかえす
    5 ほうごかす
    0.1 びょうまつ
  かくす
```

**4** 秒数を待つ命令は「せいぎょ」は「びょうまつ」です。

これを「ずっと」の最初に挿入します。

秒数は5ぐらいにしておきましょう。



```
ゲームスタート ▾ をうけとったとき
もし ステージナンバー = 2 なら
  ずっと
    5 びょうまつ
    おおきさを 50 % にする
    xざひょうを -200、yざひょうを 0 にする
    ひょうじする
    <xざひょう > 200 までくりかえす
    5 ほうごかす
    0.1 びょうまつ
  かくす
```

### じょうきゅうへん Ⅲ. 上級編

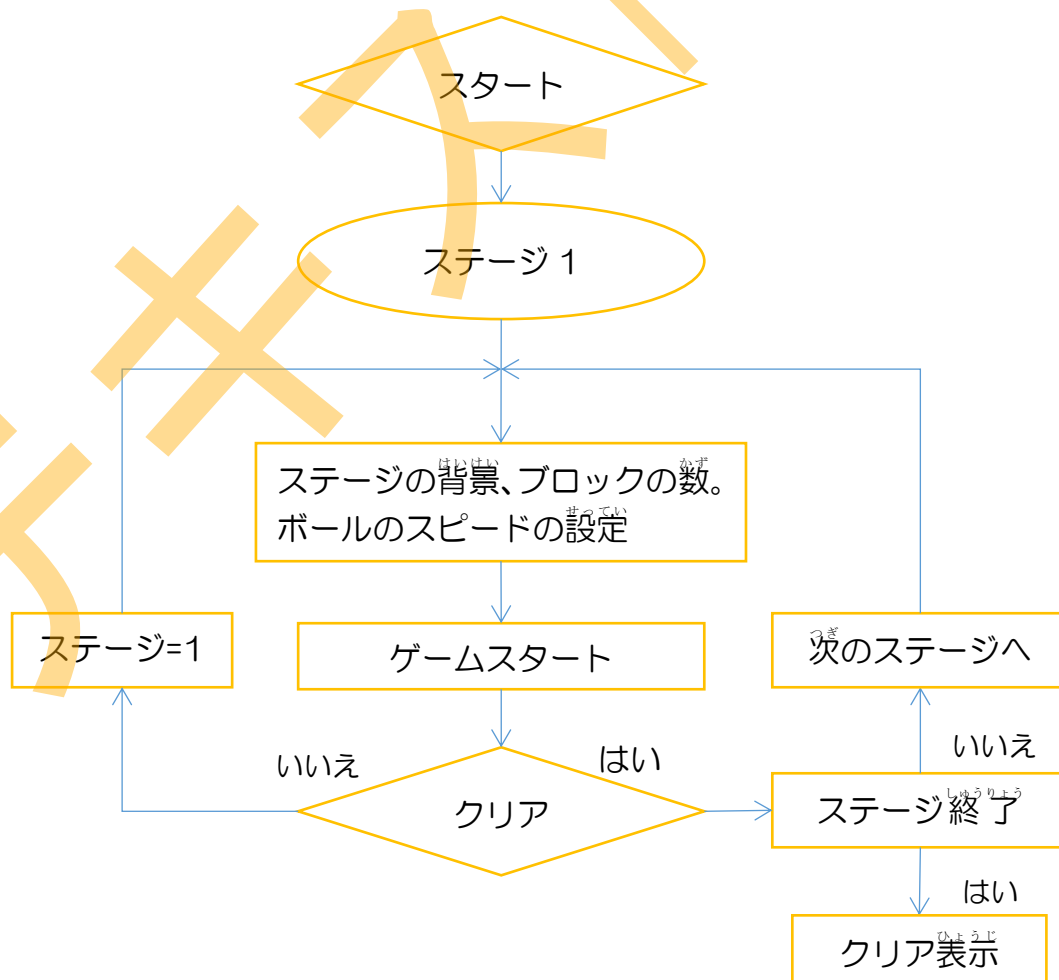
## 1. ゲームの<sup>なが</sup>流れを<sup>つく</sup>作ってみよう

これまではステージを用意して、ステージごとに背景やブロックの数、ボールの速さを変えていました。ここではゲームの流れを作ってみましょう。

例えば、ステージ 1 をクリアできたら、ステージ 2 に移動して、ミスをしたらステージ 1 からやり直しになるというものです。

他にはステージを 1 つ、増やしてみたり、全てのステージをクリアした時にクリアしたという画面を表示させたりしてみましょう。

まずはゲームの流れとともに何を挿入していくのかを説明します。ゲームの流れは下の図のようになります。

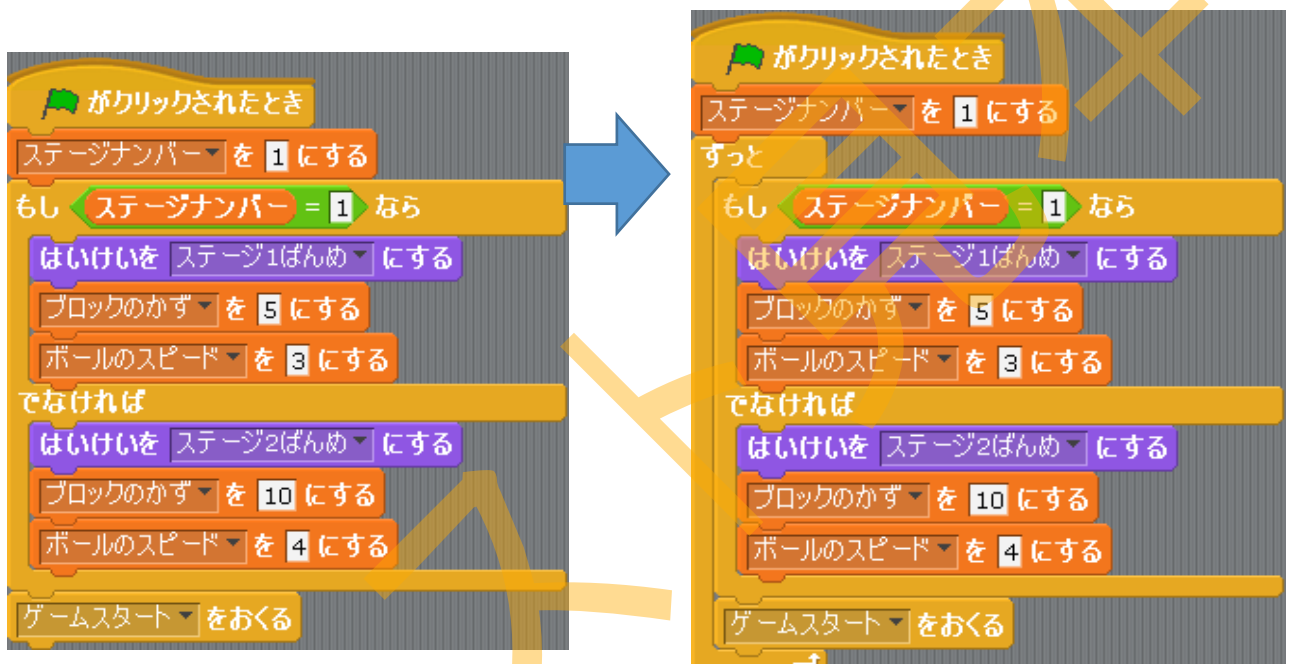


この流れを見ると繰り返していることがわかります。

したがって、まず、ステージのプログラムに「せいぎょ」から「ずっと」を挿入します。

これでゲームの流れが繰り返されるようになります。

ゲームの流れでまだ作成できていないところは「ゲーム終了」と「ステージの移動」です。



## ● ゲーム終了

1 ゲーム終了をしてからステージの移動の命令を出しますが、ゲームをしている間は待たなくてはなりません。

したがって、「せいぎょ」から「までまつ」を「ゲームスタート」の後に挿入します。

条件は当然、「ゲーム終了した」にしたいところですがそのようなパーツはありませんのでどのような時にゲームが終了したのかを考えましょう。





2 ゲームが終了したと考えられるケースはブロックの残りが0になった時とボールが下に落ちてしまった時です。ブロックの残りが0になった時を表すのは「ブロックの数」という変数が0、すなわち **ブロックのかず = 0** です。

しかし、「ボールが下に落ちてしまった」はどのように表せばいいのでしょうか？

そこでそれを表すための変数を用意しましょう。

変数の名前は「ミス」にしておきます。

この「ミス」という変数はこのように使い分けます。

- 「ミス」が0であればボールが落ちていない
- 「ミス」が1であればボールが落ちた

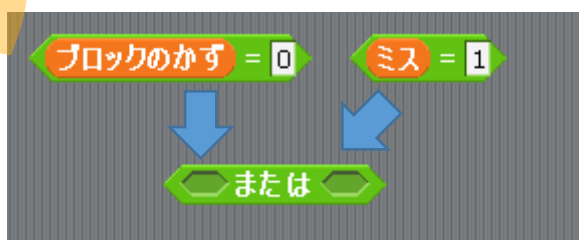
このように考えるともう1つの条件は **ミス = 1** になります。

**ブロックのかず = 0 または ミス = 1**

このどちらかの条件が満たしていることを表すのに使うのが「えんざん」にある「または」です。

下の図のように挿入しましょう。

これを条件に挿入します。



This is a screenshot of the Scratch '変数' (Variables) menu. It shows a grid of variable types: 'うごき' (Motion), 'せいぎょ' (Control), 'みため' (Appearance), 'しらべる' (Sound), 'おと' (Sound), 'えんざん' (Mathematics), 'ペン' (Drawing), and 'へんすう' (Variables). Below this is the 'あたらしいへんすうをつくる' (Create new variable) section, where 'へんすうをさくじよ' (Choose variable type) is selected. Underneath, there are checkboxes for 'ステージナンバー' (checked), 'ブロックのかず' (unchecked), 'ボールのスピード' (unchecked), and 'ミス' (checked). The bottom part of the screenshot shows the 'へんすう' (Variables) section with various operators like '+', '-', '\*', '/', and comparison operators like '<', '=', '>', along with logical operators 'かつ' (AND), 'または' (OR), and 'ではない' (NOT).

```

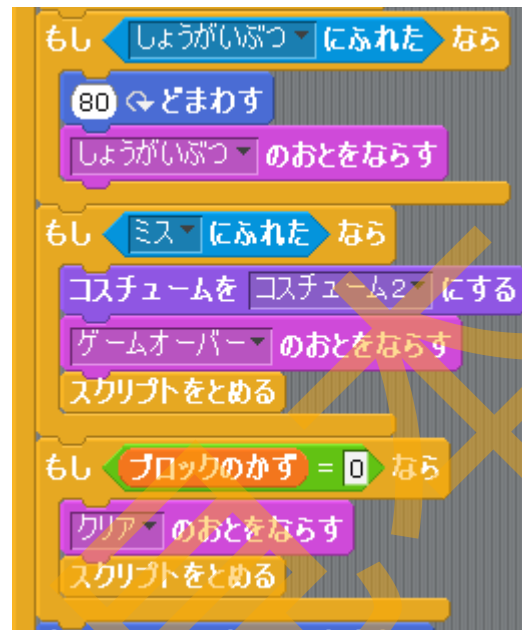
    がクリックされたとき
    ステージナンバー を 1 にする
    ずっと
    もし ステージナンバー = 1 なら
        はいけいを ステージ1ばんめ にする
        ブロックのかず を 5 にする
        ボールのスピード を 3 にする
    でなければ
        はいけいを ステージ2ばんめ にする
        ブロックのかず を 10 にする
        ボールのスピード を 4 にする
    ゲームスタート をおくる
    ブロックのかず = 0 または ミス = 1 までまつ
  
```

③ 「ミス」が1になる時はあくまでもボールが落ちた場合です。ゲームをしている時は「ミス」は0にしておく必要があります。それはゲームを始める前、つまり「ゲームスタートをおくる」の前に設定しなくてはなりません。したがって、「へんすう」から「ミスを0にする」を挿入します。それではどこで「ミスを1にする」を挿入すればいいのでしょうか？

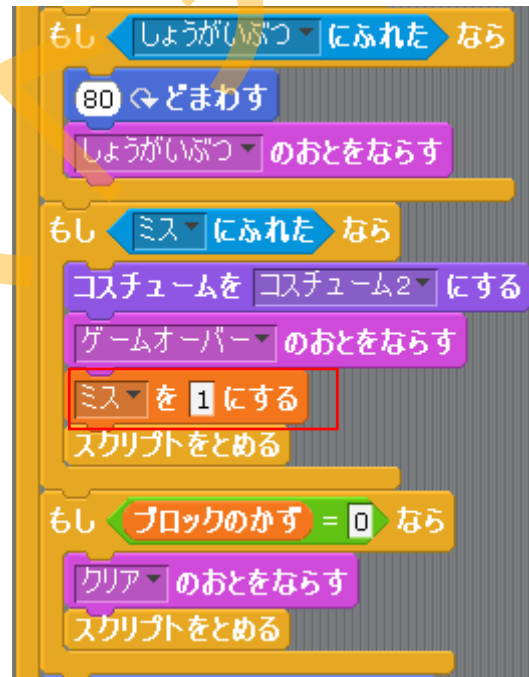
```

    がクリックされたとき
    ステージナンバー を 1 にする
    ずっと
    もし ステージナンバー = 1 なら
        はいけいを ステージ1ばんめ にする
        ブロックのかず を 5 にする
        ボールのスピード を 3 にする
    でなければ
        はいけいを ステージ2ばんめ にする
        ブロックのかず を 10 にする
        ボールのスピード を 4 にする
    ミス を 0 にする
    ゲームスタート をおくる
    ブロックのかず = 0 または ミス = 1 までまつ
  
```

**4** ボールが落ちた時の命令を出しているのはボールのプログラムです。  
「もしミスにふれたなら」がボールが落ちた時に出す命令です。  
その中に「ミスを1にする」を挿入します。



ボールのスクリプトにて  
右の図のように挿入すれば完成です。

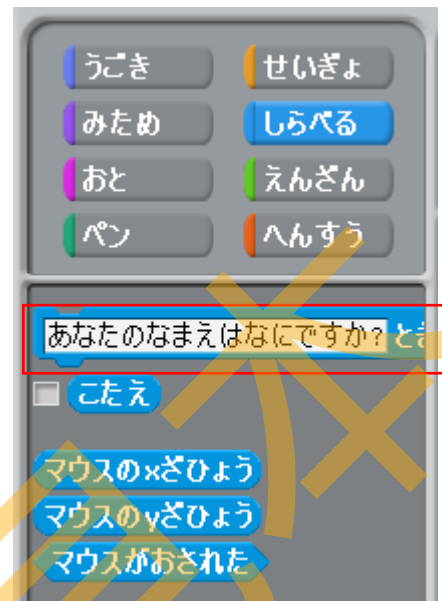


これでプログラムを動かそうと思いますがステージのプログラムで「までまつ」の後に何の命令もないと分かりにくいと思います。

その場合は「しらべる」にある「ときいてまつ」を挿入して動かしてみましよう。

右の図にある「あなたのなまえはなにですか？」をクリックして「ゲーム終了」に変えます。

この命令はメッセージを表示して、文字の入力をしますがエンターキーを押せば次の命令に進みます。



これでプログラムを動かしてみましよう。



## ● ステージの移動

ゲームが終了した後はステージを移動する命令を出します。

クリアしていれば（ブロックの数が0であれば）次のステージに進みます。

ミスしていれば（ボールが落ちれば）ステージ1からやり直しにします。

の方法をこれから説明していきましょう。

**1** クリアしたのか、ミスしたのか判定できる変数は2つあります。

それは「ブロックのかず」と「ミス」です。今回は「ブロックのかず」を使って、クリアした場合とミスした場合の命令を挿入していきましょう。

クリアした時は「ブロックの数が0」の時です。そのため、下の図のように「せいぎょ」から「もし……ならば、でなければ」を挿入して、条件を「ブロックのかず=0」にします。ここにゲームを終了した時の命令を挿入します。先程挿入した「ゲーム終了ときいてまつ」は削除しておきましょう。

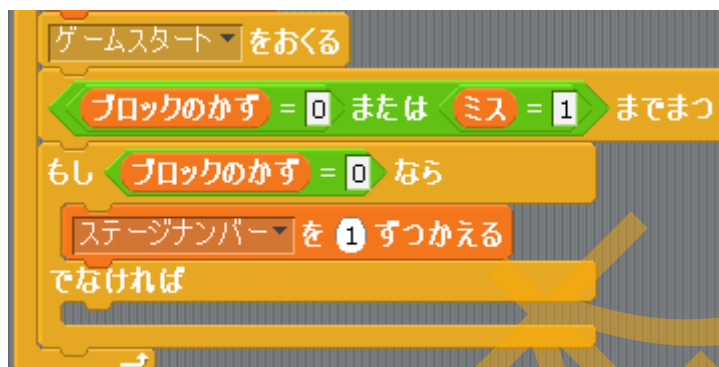
The image shows a Scratch script for stage movement. It starts with a 'when clicked' event, followed by a 'forever' loop. Inside the loop, there are two conditional blocks: 'if stage number = 1 then' and 'if not then'. The 'if stage number = 1 then' block contains: 'set stage 1 banme', 'set blocks remaining to 5', and 'set ball speed to 3'. The 'if not then' block contains: 'set stage 2 banme', 'set blocks remaining to 10', and 'set ball speed to 4'. Below these are 'set miss to 0' and 'game start' blocks. A 'wait until blocks remaining = 0 or miss = 1' block follows. Then, there are two more conditional blocks: 'if blocks remaining = 0 then' and 'if not then'. The 'if blocks remaining = 0 then' block is highlighted with a red box and has a blue arrow pointing to it from a box labeled 'クリアした時' (When cleared). The 'if not then' block is also highlighted with a red box and has a blue arrow pointing to it from a box labeled 'ミスした時' (When missed).

## 2 クリアした時の命令

を挿入します。

次（1つ先）のステージに進むということはステージナンバーが1つ増えるということです。

したがって、「へんすう」から「ステージナンバーを1ずつかえる」を挿入します。



## 3 ミスした時の命令を

挿入します。

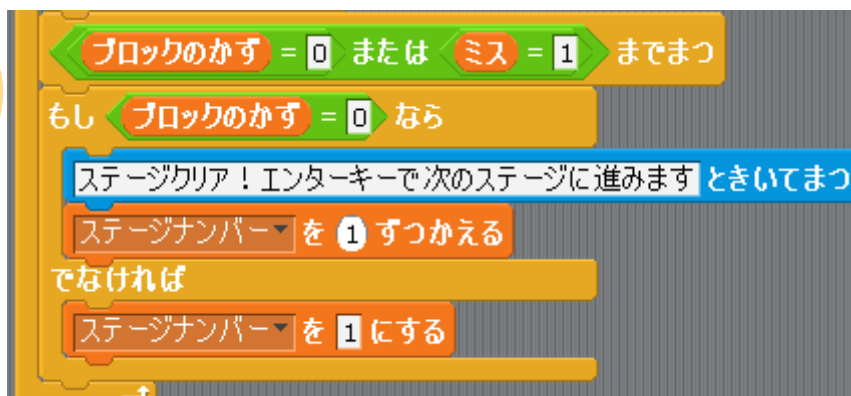
ミスした時はステージ1からやり直しますから「ステージナンバーを1にする」を挿入します。



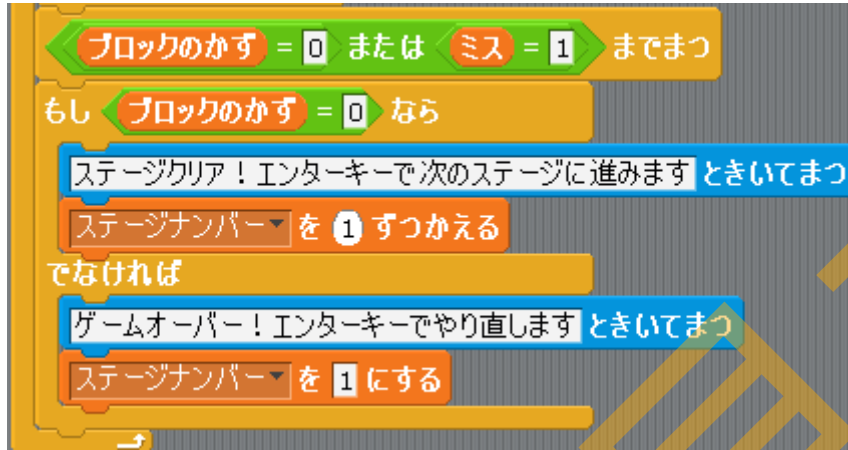
## 4 分かりやすいようにクリアした時とミスした時にメッセージを出してみましょう。

「しらべる」にある「ときいてまつ」を2つスクリプトに置きます。

1つは「ステージクリア！エンターキーで次のステージに進みます」を入力します。これをクリアした時に挿入します。



もう 1 つは「ゲームオーバー！エンターキーでやり直します」を入力して挿入します。



これでプログラムを動かしてみましよう。クリアした時やミスした時はどうなりますか？

**?** もし判定する条件が「ブロックのかず=0」ではなく、もし「ミス=1」にする場合、どのようにプログラムを変えればいいでしょう？

## ◇問題の答え◇

まず「ブロックのかず=0」の時にどこにクリアした時の命令を挿入するか、どこにミスした時の命令を挿入するかを考えます。

下の図のようになります。

このScratchスクリプトは、条件「ブロックのかず = 0 または ミス = 1」が満たされたときに実行されます。スクリプトの内容は以下の通りです。

- もし「ブロックのかず = 0」ならば
  - ステージクリア！エンターキーで次のステージに進みます
  - ステージナンバーを1ずつかえる
- でなければ
  - ゲームオーバー！エンターキーでやり直します
  - ステージナンバーを1にする

右側のボックスには、スクリプト内の特定の命令が実行されるタイミングを示しています。

- 「ステージクリア！エンターキーで次のステージに進みます」の命令は「クリアした時」に実行されます。
- 「ゲームオーバー！エンターキーでやり直します」の命令は「ミスした時」に実行されます。

では「ミス=1」にした場合はどうなるでしょうか？

このScratchスクリプトは、条件「ミス = 1」が満たされたときに実行されます。スクリプトの内容は以下の通りです。

- もし「ミス = 1」ならば
  - （この部分の命令は画像からは見えませんが、元の文脈から推測される）
- でなければ
  - （この部分の命令は画像からは見えませんが、元の文脈から推測される）

右側のボックスには、スクリプト内の特定の命令が実行されるタイミングを示しています。

- 「ミス = 1」の条件が満たされた瞬間は「ミスした時」に実行されます。
- 「でなければ」ブロック内の命令は「クリアした時」に実行されます。

上の図のように逆になっています。したがって、下の図のように挿入します。

最終的なScratchスクリプトは、条件「ブロックのかず = 0 または ミス = 1」が満たされたときに実行されます。スクリプトの内容は以下の通りです。

- もし「ミス = 1」ならば
  - ゲームオーバー！エンターキーでやり直します
  - ステージナンバーを1にする
- でなければ
  - ステージクリア！エンターキーで次のステージに進みます
  - ステージナンバーを1ずつかえる

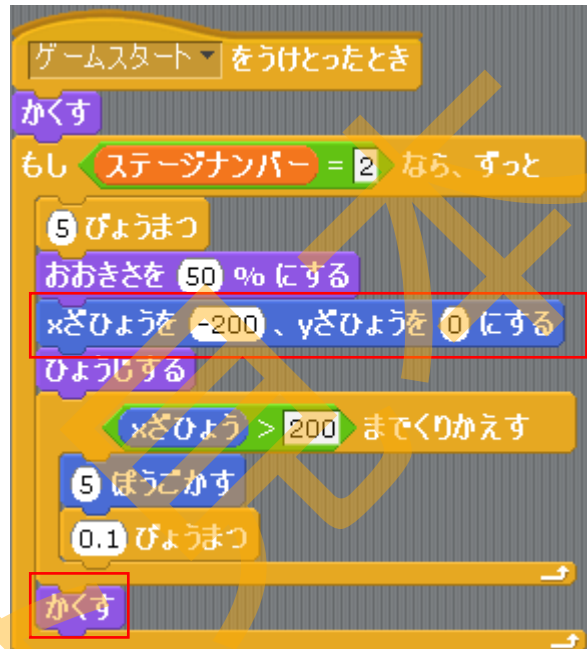
右側のボックスには、「ステージクリア！エンターキーで次のステージに進みます」の命令が「クリアした時」に実行されることを示しています。



## 2. ゲーム終了の時に障害物を消してみよう

ステージが2の時は障害物が登場しますが、これもゲームが終了した時には姿を消すようにしましょう。

右の図が障害物のプログラムになります。注目するところは「までくりかえす」です。これは右端に着くまで繰り返すという意味で、右端に着いたら繰り返しを止めて「かくす」をいう命令を出しています。この「までくりかえす」にクリアした時とボールが落ちたという条件を追加して、繰り返しを止めさせて「かくす」命令を出してみましよう。



1 挿入するところは「までくりかえす」です。

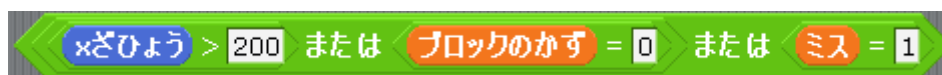
追加する条件はクリアした時、すなわち「ブロックのかす = 0」とボールが落ちた時、すなわち「ミス = 1」です。

1つでも条件を満たしていれば繰り返しを止めますから「えんざん」からは「または」を挿入すればいいのですが困ったことに条件を挿入できるスペースは2つしかありません（「または」）。

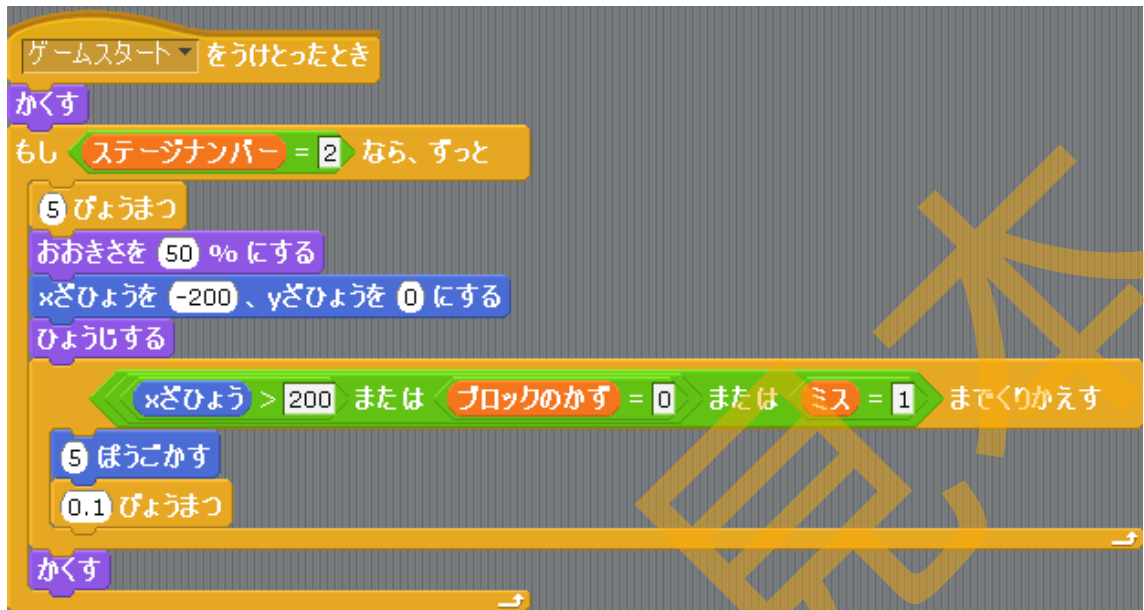
このような場合、「または」の条件に「または」を挿入します。

すると「または または」という形にこれで3つの条件が挿入できます。

すべての条件を挿入しましょう。



これを「までくりかえす」に挿入すると下の図のようになります。

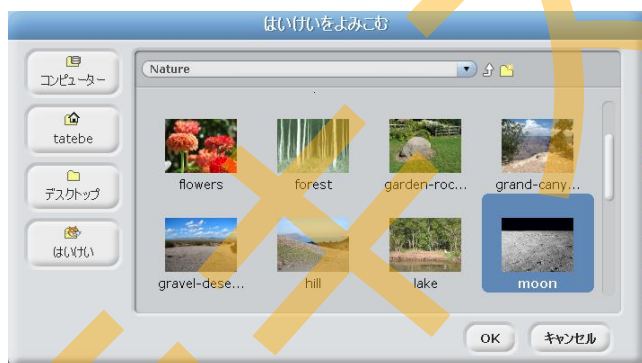


### 3. ステージを増やしてみよう

現段階でステージは2つまでしかありませんが1つ増やしてみましょう。

3つ目のステージはブロックの数を15にしてボールのスピードはステージ2よりも速くします。

まずは背景を用意します。ステージ(背景)の「はいけい」に移動して、「あたらしいはいけい」の「よみこみ」から背景を1つ追加します。



今回はNatureからMoonを選択します。

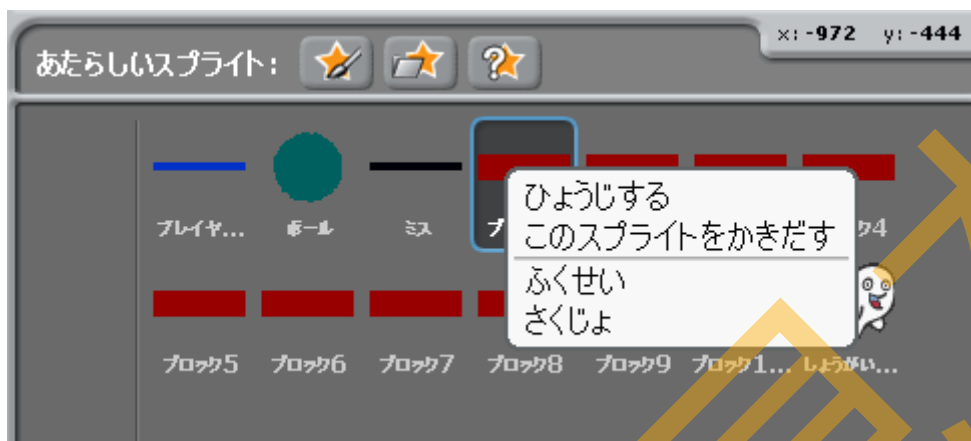


名前は「ステージ3ばんめ」にします。



次にブロックを5つ増やしましょう。

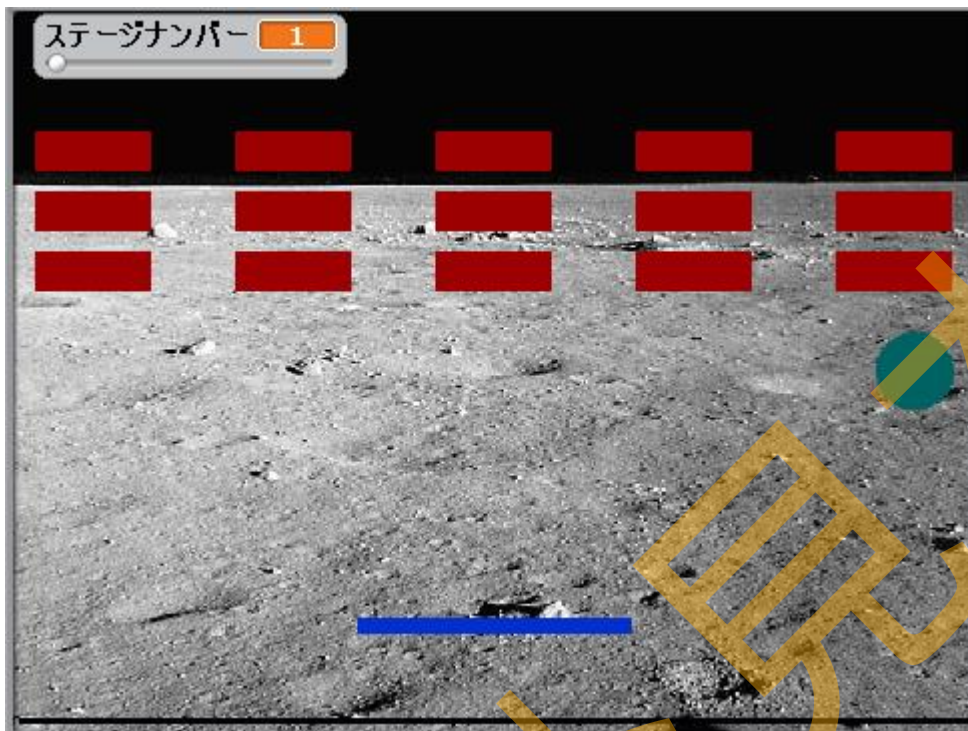
ブロックのSpriteを右クリックして「ふくせい」を選択して増やします。



名前はブロック 11、ブロック 12、ブロック 13、ブロック 14、ブロック 15 にして下の図のように並べましょう。

座標は以下のように指定します。

ブロックの名前	X座標	Y座標
ブロック 11	-200	50
ブロック 12	-100	50
ブロック 13	0	50
ブロック 14	100	50
ブロック 15	200	50



上のブロックはステージ1から真ん中のブロックはステージ2から下のブロックはステージ3から登場するようにします。  
作成が終わればプログラムを作成していきましょう。

## ● ステージのプログラム

まずはステージのプログラムの各ステージにおけるブロックの数、背景、ボールの速度を設定しているところを見ましょう。

もしもステージナンバーが3であれば、どの命令が実行されるでしょうか？

答えは「でなければ」の方です。

何故なら2の場合や3の場合でも1ではないからです。

したがって、右の図において「でなければ」の中で2であるか、3であるか場合分けしなくてはなりません。

**1** 場合分けに使うのは「せいぎよ」にある「もし……ならば、でなければ」です。

これを「でなければ」に挿入します。

条件は「ステージナンバーが2」、つまり「ステージナンバー = 2」です。



```
がクリックされたとき
ステージナンバー を 1 にする
ずっと
もし ステージナンバー = 1 なら
はいけいを ステージ1ばんめ にする
ブロックのかす を 5 にする
ボールのスピード を 3 にする
でなければ
はいけいを ステージ2ばんめ にする
ブロックのかす を 10 にする
ボールのスピード を 4 にする
ミス を 0 にする
ゲームスタート を おくる
```



```
もし ステージナンバー = 1 なら
はいけいを ステージ1ばんめ にする
ブロックのかす を 5 にする
ボールのスピード を 3 にする
でなければ
もし ステージナンバー = 2 なら
でなければ
はいけいを ステージ2ばんめ にする
ブロックのかす を 10 にする
ボールのスピード を 4 にする
```

**2** 「もしステージナンバーが 1 なら」にある「でなければ」の命令はステージが 2 の場合ですから、全て「ステージナンバーが 2 なら」に移動させます。そして、「ステージナンバーが 2 なら」にある「でなければ」にステージが 3 の場合の命令を挿入します。

```

when clicked
  stage number を 1 にする
  ずっと
    もし <ステージナンバー = 1> なら
      はいけいを ステージ1ばんめ にする
      ブロックのかず を 5 にする
      ボールのスピード を 3 にする
    でなければ
      もし <ステージナンバー = 2> なら
        はいけいを ステージ2ばんめ にする
        ブロックのかず を 10 にする
        ボールのスピード を 4 にする
      でなければ
  
```

**3** ステージ 3 では背景は「ステージ 3 ばんめ」です。ブロックの数は 15 です。そして、ボールのスピードはステージ 2 が 4 ですから 5 にしましょう。したがって、右の図のようになれば完了です。

```

when clicked
  stage number を 1 にする
  ずっと
    もし <ステージナンバー = 1> なら
      はいけいを ステージ1ばんめ にする
      ブロックのかず を 5 にする
      ボールのスピード を 3 にする
    でなければ
      もし <ステージナンバー = 2> なら
        はいけいを ステージ2ばんめ にする
        ブロックのかず を 10 にする
        ボールのスピード を 4 にする
      でなければ
        もし <ステージナンバー = 3> なら
          はいけいを ステージ3ばんめ にする
          ブロックのかず を 15 にする
          ボールのスピード を 5 にする
        でなければ
  
```

## ● ブロックのプログラム

ブロックのプログラムも変更しなければなりません。

変更するブロックは真ん中の段のブロックになります。

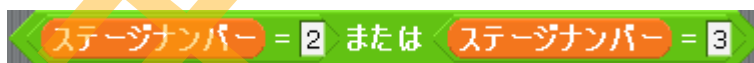


真ん中の段のブロックのプログラムは右の図のようになっています。

このブロックはステージナンバーが2の時に「ひょうじする」になっていますが3の時にも「ひょうじする」にしなくてはなりません。



ステージナンバーが2と3の時ということを表すには、

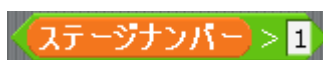


も間違いではありません。しかし、今回は別の方法を考えます。

2と3も1よりも大きな数字です。

その場合は「えんざん」に「>」を使いましょう。

開いている方が大きい数字という意味ですからこのようにしましょう。



これを条件に挿入します。





真ん中の段のブロック全てに  
この「えんざん」を条件に挿入し  
ます。

これでプログラムを動かしてステ  
ージナンバーが3の時に真ん中の  
ブロックが登場するかを確かめ  
ましょう。



次は下のブロックのプログラムです。



下のブロックはステージが3の時の  
み登場させます。ですから、条件は  
「ステージナンバー = 3」にすればい  
いのですが真ん中のブロックと同じ  
く「>」で考えると2よりも大きい  
のですから「ステージナンバー > 2」  
です。これを挿入しましょう。

これでステージ3の時に下のブロッ  
クが登場するかを確かめてみましょう。

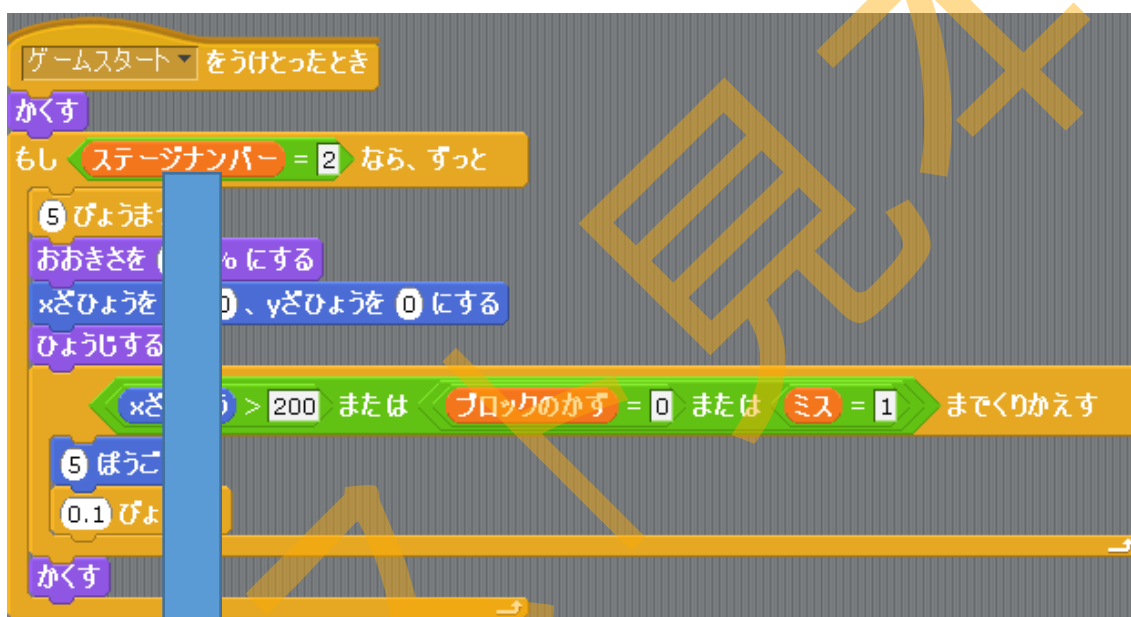


## ● しょうがいぶつ 障害物のプログラム

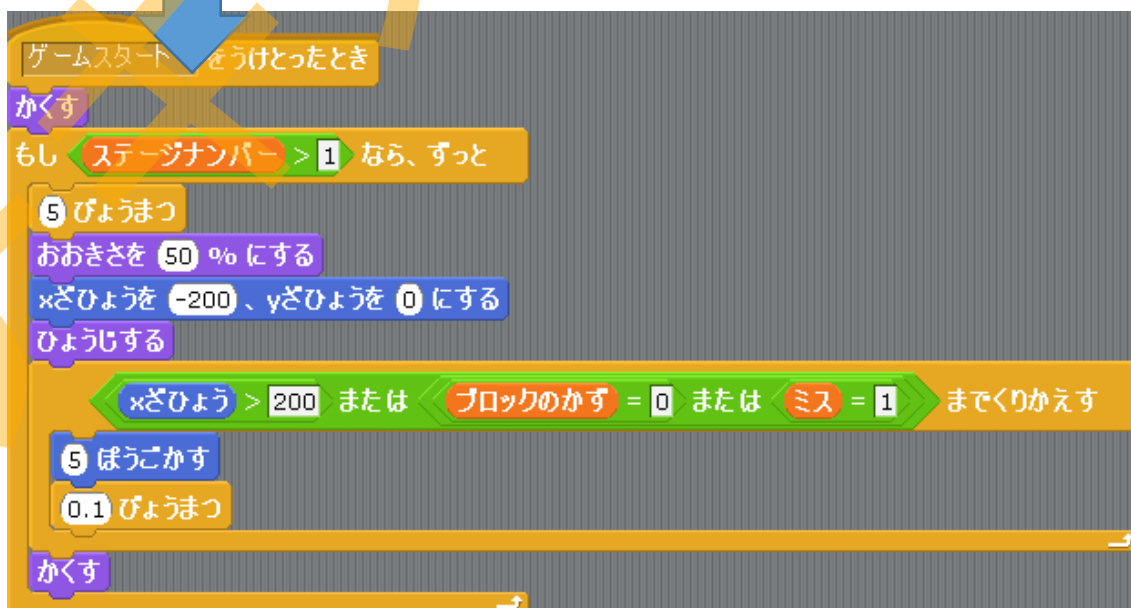
中級で作成した障害物はステージ3でも登場させましょう。

挿入する条件はブロックの場合も同じです。ステージナンバーが1より大きい

数字が条件になりますから **ステージナンバー > 1** になります。



```
ゲームスタート をうけとったとき
かくす
もし ステージナンバー = 2 なら、ずっと
  5 びょうまつ
  おおきさを ( ) % にする
  xざひょうを ( )、yざひょうを ( ) にする
  ひょうじする
  <xざひょう > 200 または <ブロックのかす = 0 または ミス = 1 > までくりかえす
  5 ぼうご
  0.1 びょうまつ
かくす
```



```
ゲームスタート をうけとったとき
かくす
もし ステージナンバー > 1 なら、ずっと
  5 びょうまつ
  おおきさを (50) % にする
  xざひょうを (-200)、yざひょうを (0) にする
  ひょうじする
  <xざひょう > 200 または <ブロックのかす = 0 または ミス = 1 > までくりかえす
  5 ぼうごかす
  0.1 びょうまつ
かくす
```

## しょうがいぶつ ぶ 4. 障害物を増やしてみよう


ステージ3では障害物を増やしましょう。

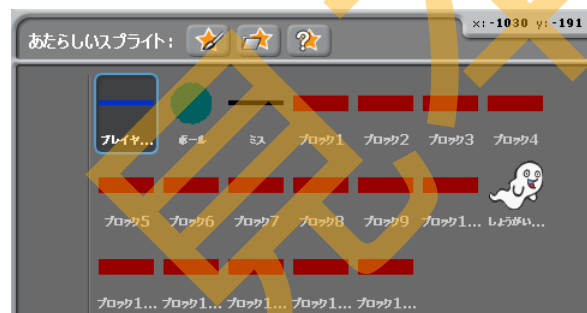
また、障害物が登場するタイミングを一定の秒数にせず、不規則にしてみましょう。

中級の第IIでは左端から右端に移動する障害物を作成しましたが、今回は右から左に移動して少しずつ下に降りていくような障害物を作成してみましょう。

スプライトを1つ用意します。

ここではスプライトをファイルから読み込みます。

あたらしいスプライトの  をクリックします。



今回は Transportation の airplane

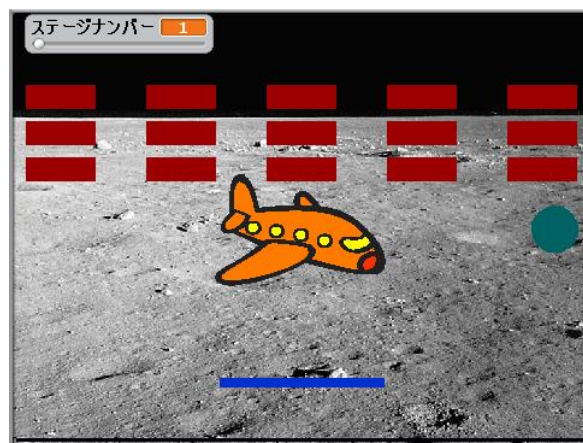
1 を選択します。



ステージに新たな障害物が登場しました。

名前は「しょうがいぶつ 2」にしておきましょう。

これもプログラムで小さくしましょう。




まずは向きを決めておきましょう。

今回は右側から左側へ少しずつ下降させていきます。

そのため、向きは-110にします。

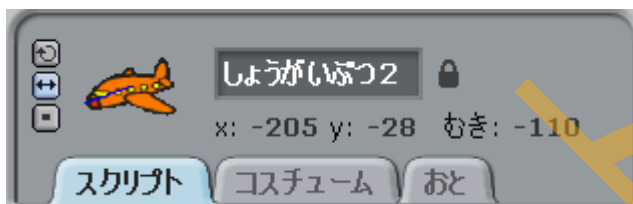
ブロックパレット上で **-110** どのむける をクリックして向きをかえます。

ここでプログラムを動かすとスプライトがひっくり返ってしまいます。

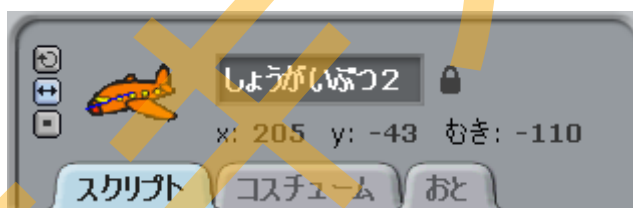
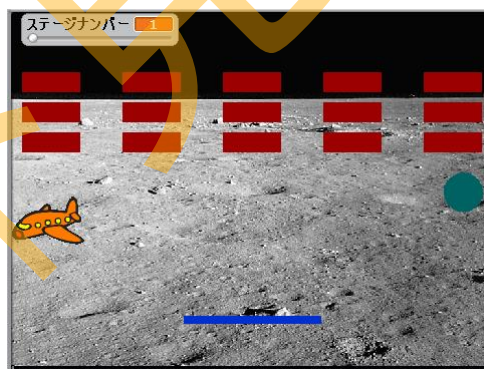
その場合は  を押して左右反転にするだけにしましょう。



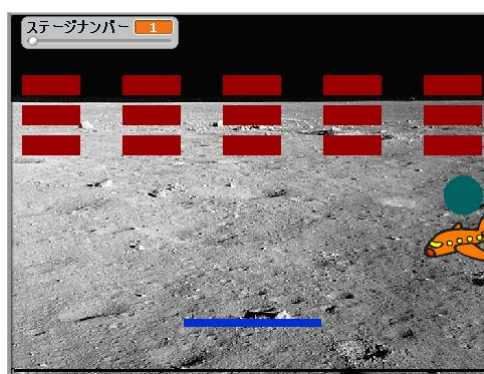
大きさを 50% にして左端と右端にスプライトがある時の x の座標を求めます。



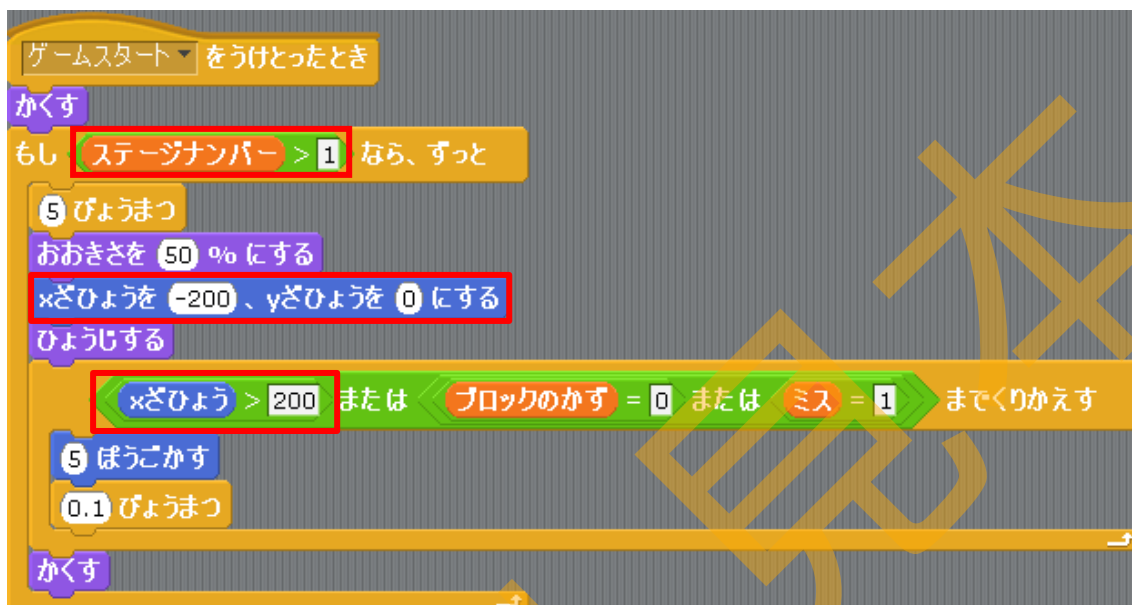
左側の x 座標は -205 にします。



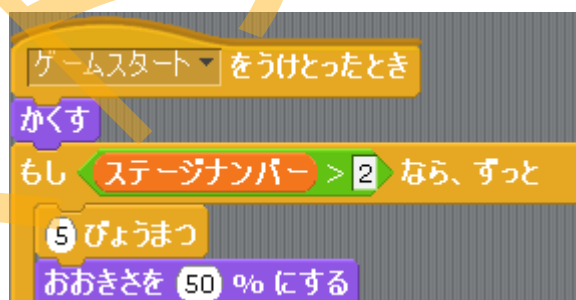
右側の x 座標は 205 にします。



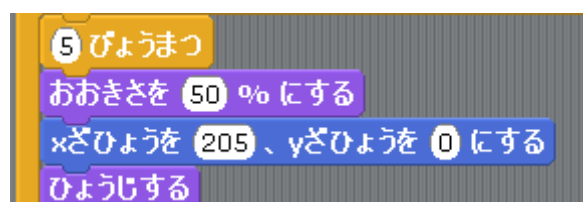
ここからプログラムの作成になりますが基本的には最初に作成した障害物のプログラムと同じです。変える場所は下の赤い枠に囲まれている箇所です。



1 今回作成している障害物はステージ3の時に登場するようにします。したがって「ステージナンバー>2」に変えます。



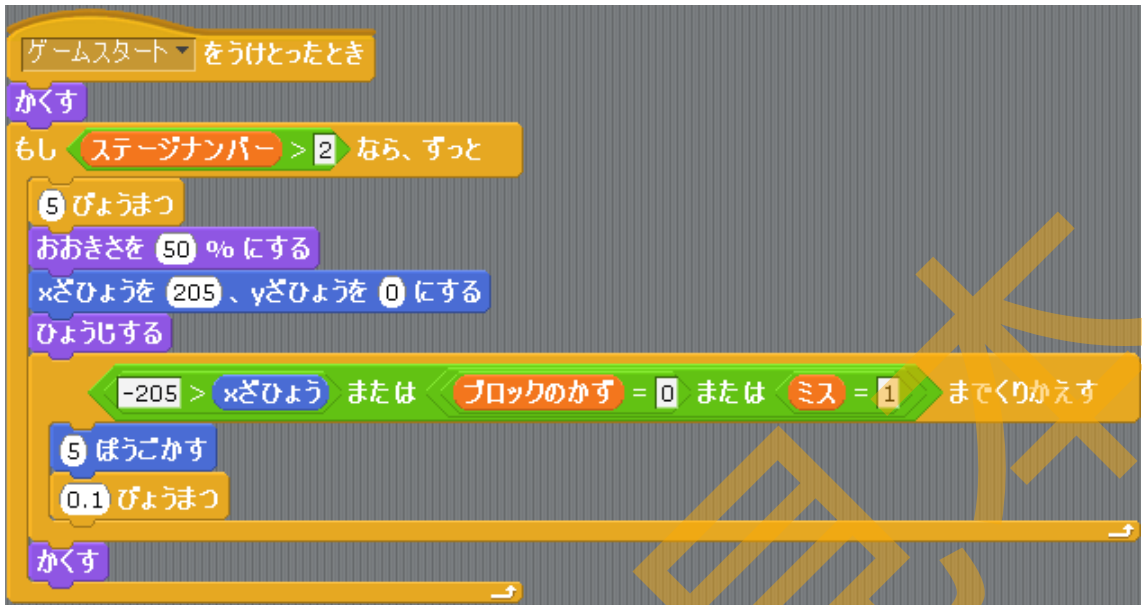
2 次に障害物を登場する場所です。つまり、「ひょうじする」の前にある命令のx座標を変えなくてはなりません。今回



は右から登場するようにしますからxの座標は205です。

3 最後に「までくりかえす」の条件を変えます。条件は「左端に着くまで」です。x座標の数字は左にあればあるほど小さくなります。したがって「左に着く」とはx座標の数字が左端の座標よりも小さくなった場合です。

よって `-205 > x座標` になります。



このようになっていれば完成です。

次はボールのプログラムでこの障害物に当たった時の命令を挿入しましょう。

**4** ボールのプログラムにはすでに最初に作成した障害物が当たった時の命令が挿入されています。今回、作成した障害物が当たった場合でも、同じような命令を挿入します。



右の図のように挿入してプログラムを動かして確認しましょう。

障害物が当たった時にボールの進行方向は変わっていますか？

```
もし <プレイヤー> にふれた なら
  80 秒 まで 待たす
  打ち返し のおとをならす
もし <いろにふれた> なら
  80 秒 まで 待たす
もし <しょうがいぶつ> にふれた なら
  80 秒 まで 待たす
  しょうがいぶつ のおとをならす
もし <しょうがいぶつ2> にふれた なら
  80 秒 まで 待たす
  しょうがいぶつ のおとをならす
もし <ミス> にふれた なら
  コスチュームを コスチューム2 にする
  しょうがいぶつ のおとをならす
  ミス を 1 にする
  スクリプトをとめる
もし <ブロックのかず = 0> なら
  クリア のおとをならす
  スクリプトをとめる
もしはしについたら、はねかえる
```

## ● 登場するまでの時間を不規則にする

これまで「障害物が端についてから消えてまた登場する」には「びょうまつ」の命令でまた登場するまでの時間を設定していました。

つまり、今までは障害物は端についてから5秒たつとまた登場するようになっていたのです。

これを3秒、5秒、8秒、6秒のように不規則にできればゲームが面白くなると思われれます。

これを実現させるために使うのが乱数です。乱数とは指定した範囲でランダムで発生させた数字です。

乱数は「えんざん」の中にあります。

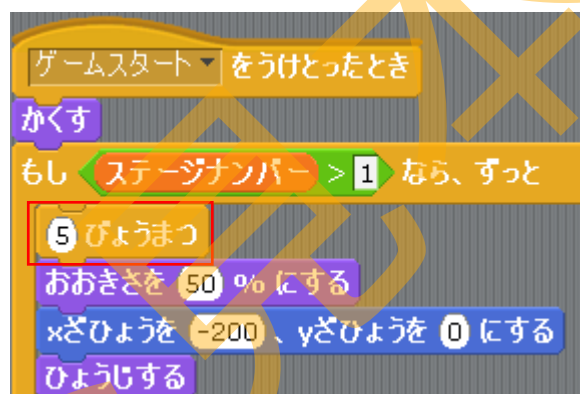
「えんざん」には「1から10までのらんすう」があります。これを使うと1から10の間の数字がランダムに発生します。

ですから、最初に2になっても、次が必ず2になるわけではなく、3や5などになることもあります。

これを使えば不規則に障害物を登場させることができます。

「1から10までのらんすう」は変数と同じようにマウスをクリックしてドラッグすれば「びょうまつ」に挿入できます。

障害物すべてに挿入します。



1から10までのらんすう

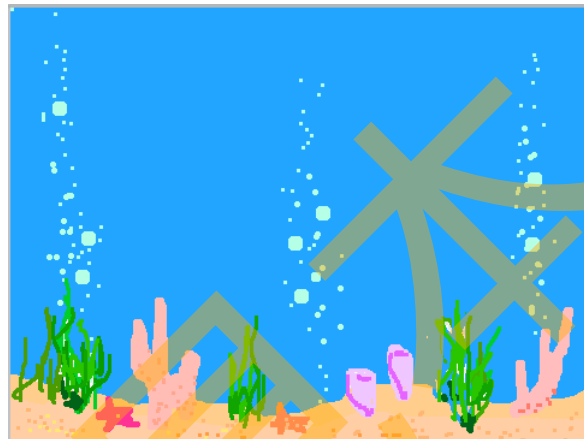




## 5. 背景にアニメーションを入れてみよう

これまでステージごとに背景を違うものにしていましたが背景に簡単なアニメーションをつけてみましょう。

今回はステージ2の背景にアニメーションをつけてみましょう。



1 上の図を一から真似していると

大変時間がかかります。

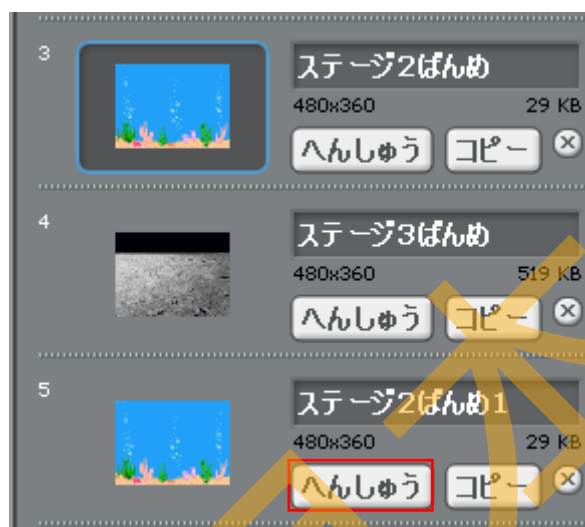
ですから、背景をコピーしてみましょう。

「コピー」をクリックすれば背景がコピーされます。




2 背景を少し変えてみましょう。

「へんしゅう」をクリックすればペイントエディタの画面が表示されます。

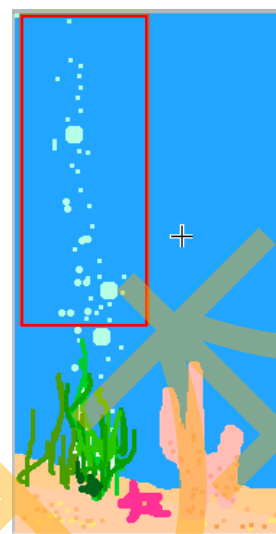


3 変更しやすいところ、例えば泡を変えてみましょう。



例えば、泡があるところを  で囲む（右の図）とその部分をマウスで移動することができます。



下の図ではそれぞれの泡の  で囲んで上に移動させました。




色がなくなった場所には水と同じ色を塗ります。



をクリックして、青いところにマウスを移動させてクリックしてみましょう。

するとパレットの横にある  が  に変わります。

これで  で色を塗ってみましょう。



色を塗った場所に泡を描いてみましょう。



これで背景が完成です。

次にプログラムで背景を切り替えて表示してみましょう。

背景の名前は「ステージ 2 ばんめ 1」になります。

背景を切り替えることができるのはステージのプログラムです。  
したがって、ステージのスク립トでプログラムを作成します。

**1** はいけい背景のアニメーションはゲームがスタートして  
から始めます。  
したがって、最初に「ゲームスタートをうけとった  
とき」を置きます。



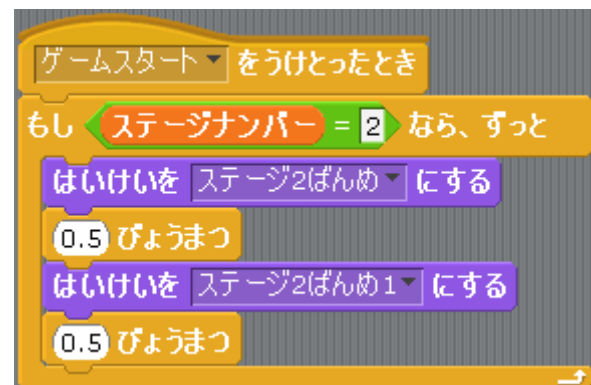
**2** 次にアニメーションにする方法を  
考えます。

今回は 1枚目の画像を表示、0.5秒待つ、  
2枚目の画像を表示、0.5秒待つ……の  
繰り返しで行います。



今回はステージ2の背景のみにアニメーションを設定します。  
したがって「せいぎょ」から「もし……なら、ずっと」を挿入します。  
条件は「ステージナンバー=2」です。

**3** 「はいけいをステージ2ばんめにする」、  
「0.5 びょうまつ」、  
「はいけいをステージ1ばんめにする」、  
「0.5 びょうまつ」を挿入します。



## 6. 全てのステージのクリア演出を入れてみよう

ステージは全てで3つあります。

この3つのステージを全てクリアした場合にクリアを祝福するような演出を入れてみましょう。

まずはその演出のための背景を用意します。

今回はIndoorにある  
スポットライト ステージ  
spotlight\_stageを使います。  
名前は「ゲームクリア」にし  
ます。



この画像だけではクリアを祝福する  
ような雰囲気足りないので、この  
画像の中央に「おめでとう」の文字を  
挿入してみましょう。

「へんしゅう」をクリックしてペ  
イントエディタを開きましょう。



T

をクリックすると画像に文字が入力できるようになります。

画像に左の図のようなものが表示されます。  
ここから入力した文字が表示されるよう  
になります。



「おめでとう」を入力してみましょう。

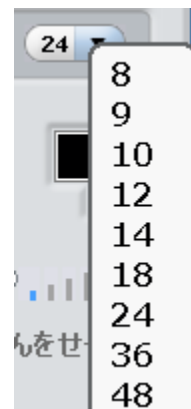


「おめでとう」が表示されましたが、文字が小さいので少し大きく  
 しましょう。

上の図の赤い枠で調整することができます。

この数字は大きければ大きいほど、文字は大きくなります。

現在は「24」ですが1つ大きめの「36」にしてみましょう。



文字が大きくなりましたが中央  
 の位置より少しずれています。

中央に調整しましょう。

黒い四角をマウスでクリックし  
 てドラッグしながら移動させま  
 す。



次に文字の色を変えてみましょう。

**T**を選択している時にパレットの色をクリックすれば文字の色が変わります。

ブロックの色と同じにします。



これで背景は完成です。

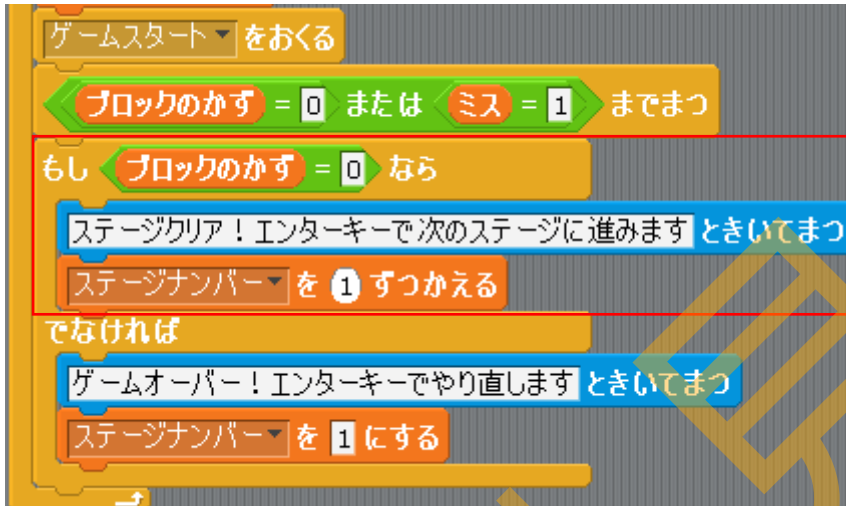


これで背景の準備はできましたが、これをプログラムに組み込ませるにはどうすればいいかを説明しましょう。



1 クリアした時の演出はステージのプログラムで行います。

ゲームを終えて、クリアした時に最後、つまり、ステージが3であれば命令を出します。



まずはステージのプログラムでクリアした時（ブロックのかず=0）に「せいぎよ」から「もし……ならばでなければ」を挿入します。



条件は「ステージナンバー=3」にします。

```
ブロックのかず = 0 または ミス = 1 までまつ
もし ブロックのかず = 0 なら
  もし ステージナンバー = 3 なら
    でなければ
      ステージクリア！エンターキーで次のステージに進みます ときいてまつ
      ステージナンバー を 1 ずつかえる
    でなければ
      ゲームオーバー！エンターキーでやり直します ときいてまつ
      ステージナンバー を 1 にする
```

2 「みため」から「はいけいをゲームクリアにする」を挿入します。

```
ブロックのかず = 0 または ミス = 1 までまつ
もし ブロックのかず = 0 なら
  もし ステージナンバー = 3 なら
    はいけいをゲームクリアにする
    でなければ
      ステージクリア！エンターキーで次のステージに進みます ときいてまつ
      ステージナンバー を 1 ずつかえる
    でなければ
      ゲームオーバー！エンターキーでやり直します ときいてまつ
      ステージナンバー を 1 にする
```

次にメッセージを出してみましょう。

「おめでとう！再挑戦する場合はエンターキーを押してください」というメッセージにします。

```

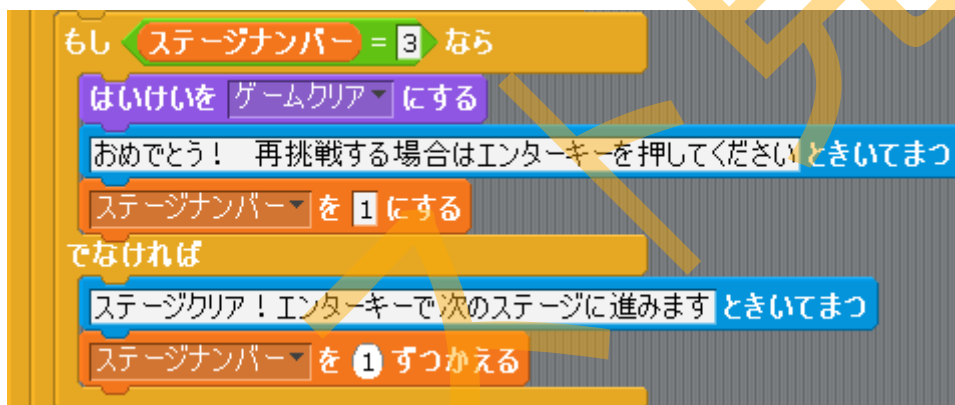
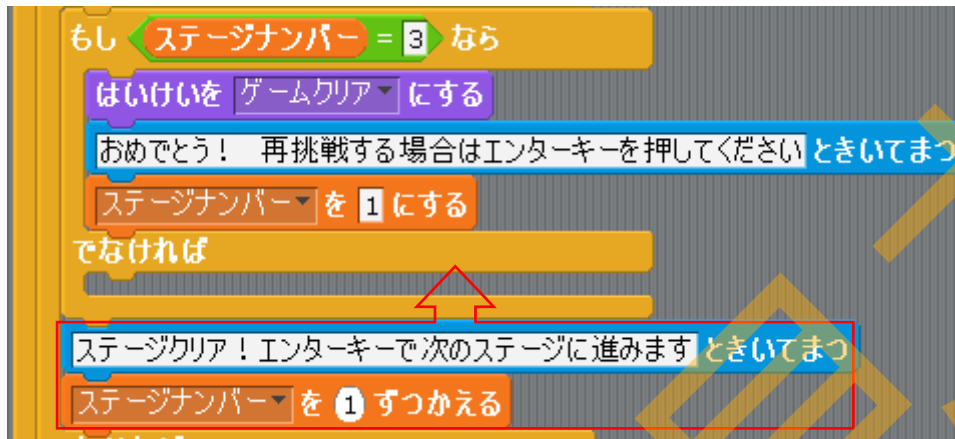
    <ブロックのかず = 0 または ミス = 1 > までまつ
    もし <ブロックのかず = 0 > なら
        もし <ステージナンバー = 3 > なら
            はいけいを ゲームクリア にする
            おめでとう！ 再挑戦する場合はエンターキーを押してください ときいてまつ
            でなければ
                ステージクリア！エンターキーで次のステージに進みます ときいてまつ
                ステージナンバー を 1 ずつかえる
            でなければ
                ゲームオーバー！エンターキーでやり直します ときいてまつ
                ステージナンバー を 1 にする
    →
  
```

最後に「ステージナンバーを 1 にする」を挿入します。これで最初からゲームができるようになります。

```

    <ブロックのかず = 0 または ミス = 1 > までまつ
    もし <ブロックのかず = 0 > なら
        もし <ステージナンバー = 3 > なら
            はいけいを ゲームクリア にする
            おめでとう！ 再挑戦する場合はエンターキーを押してください ときいてまつ
            ステージナンバー を 1 にする
        でなければ
            ステージクリア！エンターキーで次のステージに進みます ときいてまつ
            ステージナンバー を 1 ずつかえる
        でなければ
            ゲームオーバー！エンターキーでやり直します ときいてまつ
            ステージナンバー を 1 にする
    →
  
```

- ③ もともとあった「ステージクリア！エンターキーで次のステージに進みますときいてまつ」と「ステージナンバーを1にする」を「でなければ」に移動します。



これでプログラムを動かしてみましよう。

ステージ3をクリアした時に設定した背景は表示されますか？

## IV.もしもうまく動作しない場合

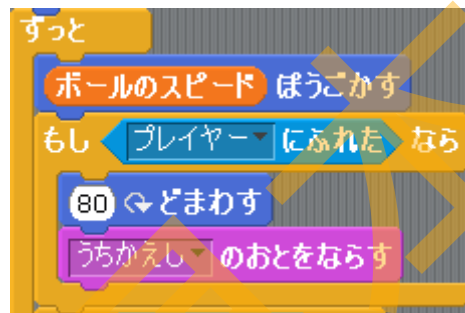
テキスト通りにプログラムを作成して、動かしてみてもうまく動作しないことがあります。ここではいくつかのケースを例に挙げて解決方法を紹介します。

- ① 「プレイヤー」を「ボール」に当てた時に「ボール」がその場で振動する場合
- ② ステージをクリアして次のステージに進んだ時、ブロックが1つない場合

# 1. 当たった時に「ボール」がその場で振動する

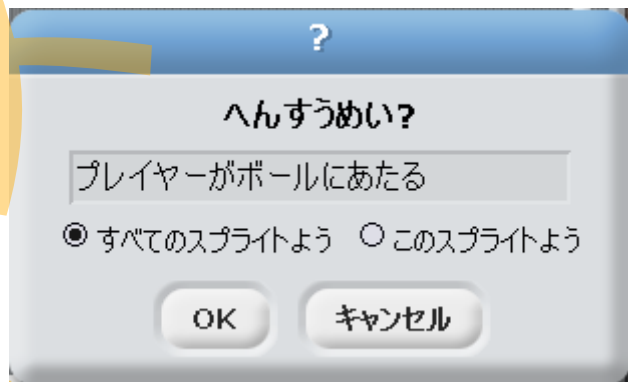
## 場合

「ボール」が「プレイヤー」に当たった時のプログラムを見ましょう。右の図がそのプログラムになります。



「プレイヤーにふれた」場合に80度右回転するようになっていますが、あたりどころや変える向きによって、「ボール」と「プレイヤー」が当たった状態が続いてしまいます。何回も何回もこの条件が満たされるとボールを何度も向きを変えてしまい、「ボール」が振動しているように見えるのです。

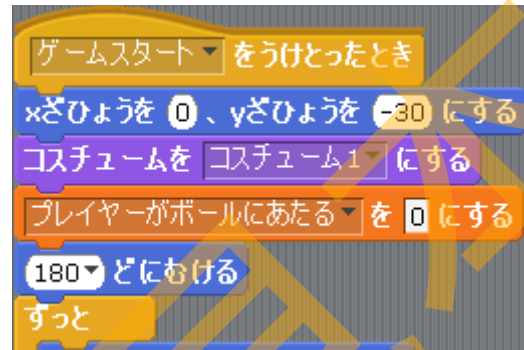
これを解決するためには、「ボール」が「ショット」に当たった場合は一度だけ「80度右回転する」を実行して、「ボール」が「ショット」から離れるまでは「80度右回転する」を実行しないようにします。



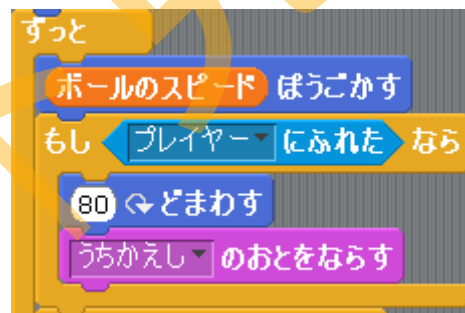
これを実行するためには新しい変数を用意しましょう。変数の名前は「プレイヤーがボールにあたる」にします。

この「プレイヤーがボールにあたる」が0の時に「プレイヤー」が「ボール」にあ  
っていない場合、1の時に「プレイヤー」が「ボール」が当たっている場合にしま  
す。

ゲームが始まった時、つまり、「ゲームス  
タートを受けとったとき」は「ボール」  
は「プレイヤー」には当たっていません  
ので「プレイヤーがボールにあたる」は  
0にしましょう。



次に「プレイヤーにふれた」ときのプログラ  
ムを変更します。条件を1つ追加しなくて  
はなりません。最初に「プレイヤーにふれた」  
場合のみ、80度右回転するようにします。



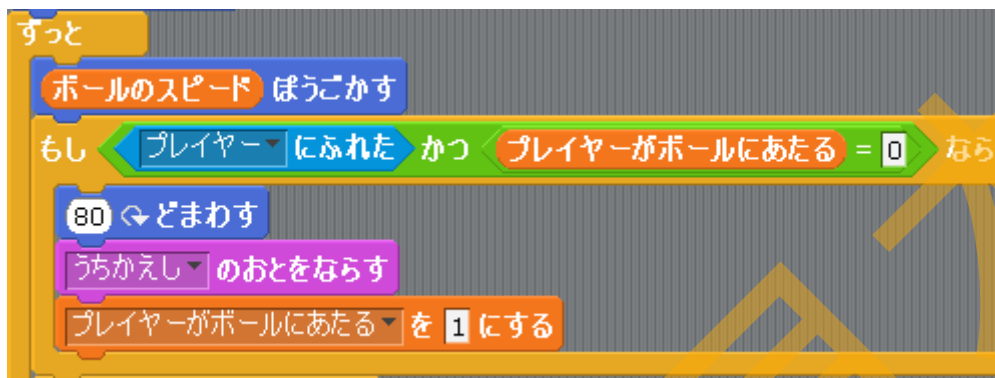
最初に「プレイヤーにふれた」時の「プレイヤーがボールにあたる」の数字は0  
です。条件は「プレイヤーにふれた」と「プレイヤーがボールにあたる」の数字が  
0を同時に満たす時ですので「かつ」を使って2つの条件を挿入します。



これが新しい条件になります。



「プレイヤー」が「ボール」が当たっている時には「プレイヤーがボールにあたる」の数字を1にしますので「プレイヤーがボールにあたるを1にする」を挿入します。



これでもう一度「プレイヤーにふれた」としても「プレイヤーがボールにあたる」が1になるので右回転の命令は実行されなくなりますので振動が起こらなくなります。

もちろん、「ボール」が「プレイヤー」にふれなくなった場合、「プレイヤーがボールにあたる」を0に戻さなくてはなりません。

そこで「もし……なら」を挿入します。



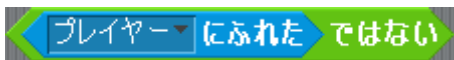
条件は「プレイヤーにふれていない」と「プレイヤーがボールにあたる」が1になっている場合です。



「えんざん」の中には「ではない」という命令があります。



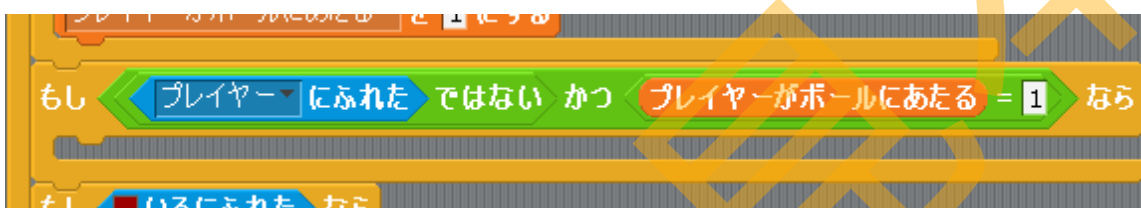
この「ではない」に「プレイヤーにふれた」を挿入しましょう。



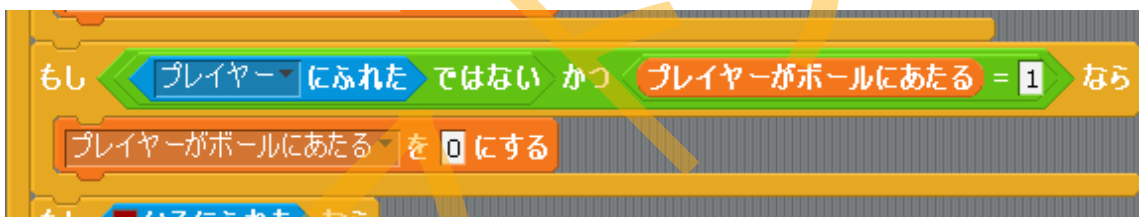
これで「プレイヤーにふれていない」という条件が完成しました。これと「プレイヤーがボールにあたる」が1という条件を「かつ」でつなぎます。



これを「もし……なら」に挿入します。



ここに「プレイヤーがボールにあたる」を0にする命令を挿入します。



これで完了です。

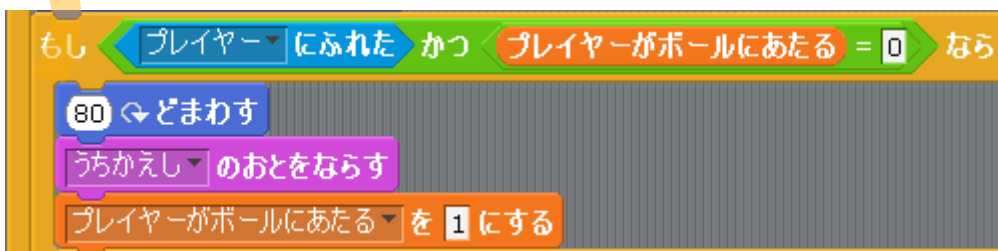
これで振動はしなくなりますが80度右回転するの命令を実行しますので、向きが下



になってしまう場合があります。

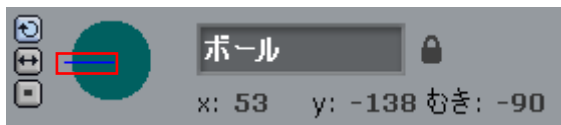
例えば、右の図の状態<sup>す</sup>で下方向にボールが進んでいると「80度右回転する」の命令を実行しないので、そのまま下に落ちてゲームオーバーになってしまいます。

そこで「ボール」が「プレイヤー」に当たった場合は必ず上の方向に進んでいくように向きを変えるようにします。下の図のプログラムに命令を追加します。



まずは上の方向になる角度の範囲を確認しましょう。

左向きは「むき」が-90 になっています。



右向きは「むき」が 90 になっています。



このことから上方方向であるのは「むき」の範囲は-90 より大きく、かつ、90 より小さい場合です。つまり、この範囲内で向きを変えるようにすればいいのです。

そこでまずは「80」を削除して「90」を挿入します。

次に「えんざん」から「1 から 10 までのらんすう」を「90」に挿入します。



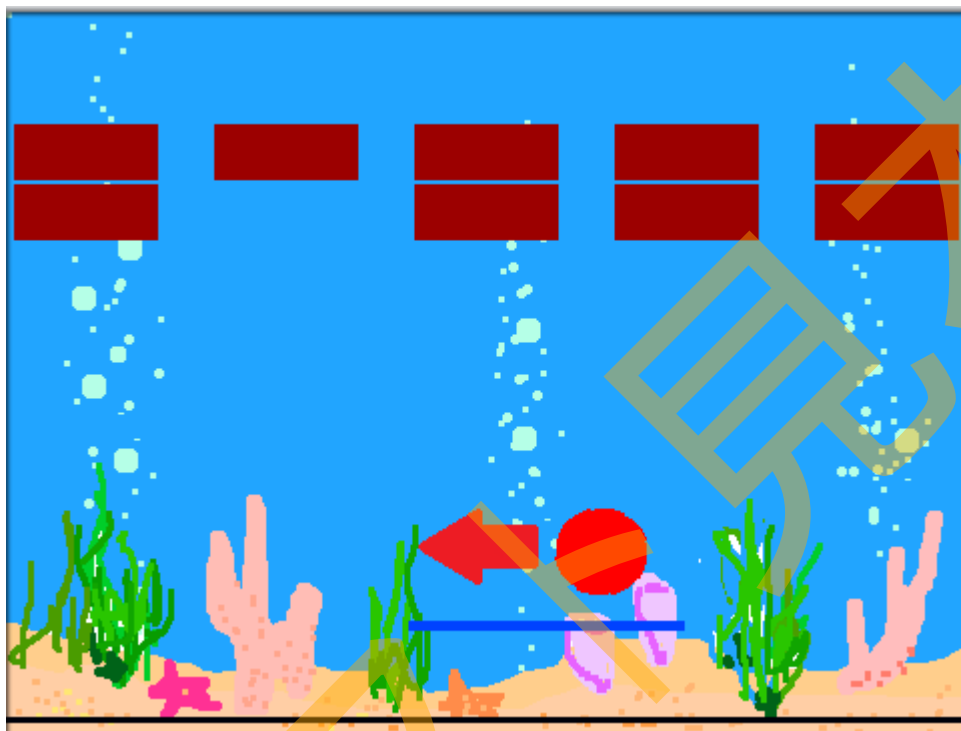
「むき」の範囲は-90 より大きいのですから-89以上になります。そして、90 より小さいのですから 89 以下になります。

つまり、「-89 から 89 までのらんすう」が発生するようにしましょう。



これでプログラムを動かしてみよう。

「-89 から 89 までのらんすう」の場合、ボールが大きく左右に動いて、ボールがなかなか真っ直ぐブロックの方向へ進まないことがあります。これは乱数でボールの向きが「-89」や「89」になるためです。



そのため、乱数を「-89 から 89 までのらんすう」から「-60 から 60 までのらんすう」に変えてみましょう。

**-60 から 60 までのらんすう** どのむける

これでボールはブロックの方向へ進みやすくなります。

## 2. 次のステージに進んだ時、ブロックが1つない場合

ステージをクリアした時に次のステージに行く時に、ステージをクリアした時に「ボール」の位置にあった「ブロック」が最初からない場合があります。

次のステージに進む時はボールを開始の位置にする命令とブロックを表示する命令は「ゲームスタートをうけとったとき」に実行しています。

ボールのプログラム

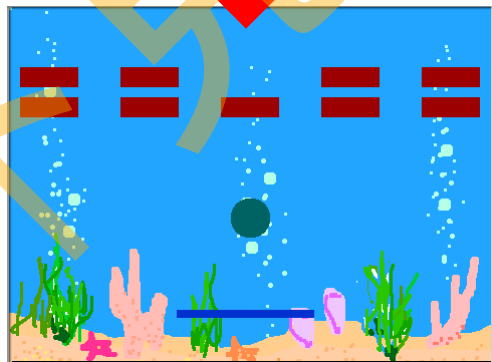
```

ゲームスタート をうけとったとき
x座ひょうを 0、y座ひょうを -30 にする
コスチュームを コスチューム1 にする
プレイヤーがボールにあたる を 0 にする
180 どのむける
ずっと
  
```

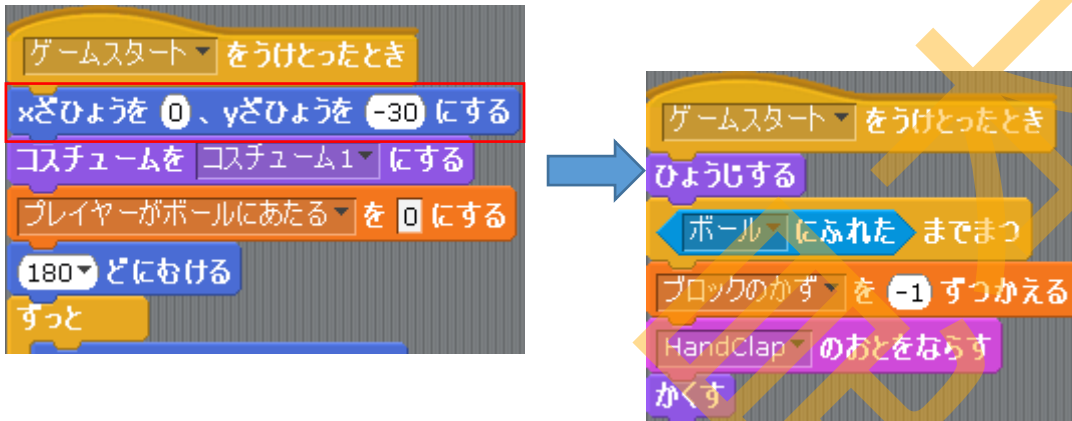
ブロックのプログラム

```

ゲームスタート をうけとったとき
ひょうじする
ボール にふれた までまつ
ブロックのかず を -1 ずつかえる
HandClap のおとをならす
かくす
  
```



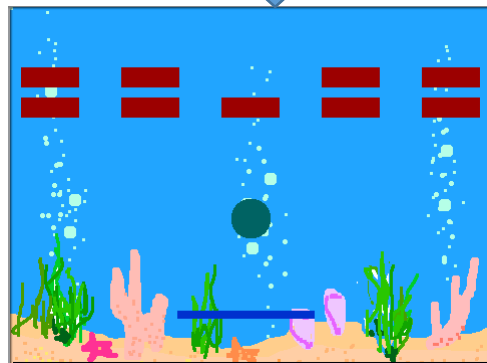
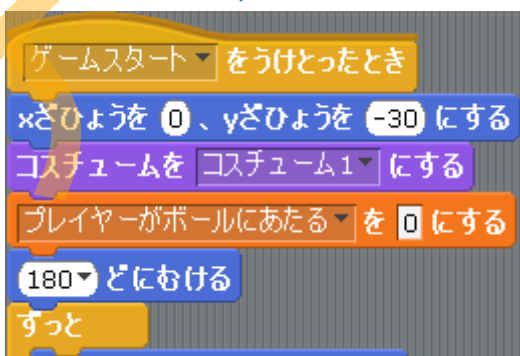
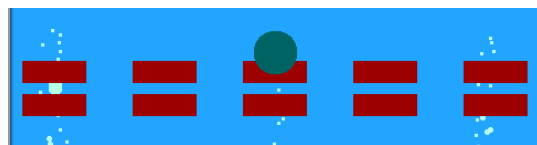
実はボールとブロックのプログラムに命令には少なからず差があり、もしもボールのプログラムが先に開始の時の座標の設定の命令を実行した後にブロックの表示の命令が実行された場合（ブロックの配置よりボールの配置が先の場合）は問題がありません。



しかし、先にブロックのプログラムの命令が実行された後にボールのプログラムが開始の時の座標の設定の命令を実行した場合（ボールの配置よりブロックの配置が先の場合）に問題が起こります。

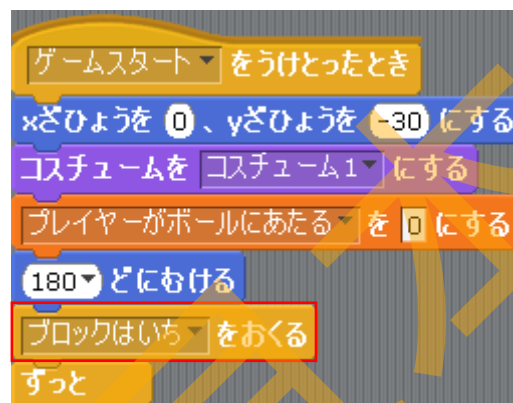


ボールにふれる位置にボールが残っているため（下の図）、「ボールにふれた」の条件を満たしてブロックが消えてしまう。



これを防ぐためにはボールを最初の位置に設定した後、ボールのプログラムからブロック全てにブロックを表示するようにメッセージを送ります。

つまり、ボールのプログラムにおいて「ずっと」の前に「せいぎよ」から「…」をおくを挿入します。メッセージの名前は「ブロックはいち」にします。必ず「xざひょうを0、yざひょうを-30にする」と「ずっと」の間に挿入しましょう。



次に全てのブロックのプログラムは「ゲームスタートをうけとったとき」に実行していますが「ブロックはいちをうけとったとき」に変更します。



これでボールを最初の位置を設定した後、ブロックの設定をします。

これでプログラムを動かしてみましよう。

三井天下見本

## プログラミング入門 ゲーム ② ブロック崩し

---

2018年12月1日 第2版

本書の複写複製(コピー)は、特定の場合を除き、著作者の権利侵害になります。

### 連絡先

(株)日本ビーコム

〒520-0802

滋賀県大津市馬場3-2-25 ワカヤマビル 2F

Tel 077-527-5681 Fax 077-527-5687



- Microsoft、Windows は、米国 Microsoft Corporation の米国およびその他の国における登録商標です。
- その他、記載されている会社名、製品名は、各社の商標および登録商標です。
- テキストに記載されている内容、仕様は予告なしに変更されることがあります。