



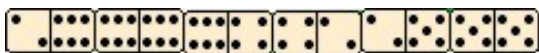
PROGRAMANDO EM

SCRATCH

Relato da génese de um projecto Scratch



INTRODUÇÃO



Programar é...

...ordenar uma série de ideias simples num conjunto coerente e funcional capaz de produzir um resultado desejável; mas é, também, o confronto permanente com a própria e muito humana capacidade de errar e de se corrigir.



Objectivo

O objectivo deste documento é o de acompanhar o desenvolvimento de um projecto Scratch, desde a concepção até à entrada em produção. É destinado a pessoas já iniciadas no ambiente Scratch que, porventura, sintam dificuldade em levar à prática as suas ideias.

Estrutura deste documento

O presente documento foi escrito à medida que o projecto “Fracções-4” se foi desenvolvendo, desde o momento da sua concepção intelectual até à sua realização no ambiente Scratch, passando pelos avanços e hesitações, pelos erros e emendas, pelos desvios ao plano inicial, pelas dificuldades e soluções de recurso mais ou menos inspiradas, enfim, por tudo o que transforma o ofício de programar numa arte.

Oxalá, que as dificuldades narradas não desesperem o leitor, em vez de o incentivar a enfrentá-las e a vencê-las, como é intenção e prática do autor.

O projecto **Fracções-4** está disponível em <http://scratch.mit.edu/projects/ffred/123985>



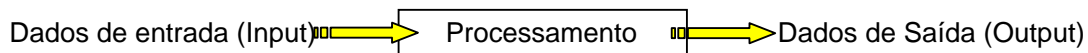
Expresso os meus agradecimentos à colaboração inestimável que **Teresa Martinho Marques** e **Rui Soares** deram à realização deste documento

ÍNDICE

1. O PROBLEMA	3
2. A IDEIA.....	3
3. OS DADOS DE ENTRADA ('INPUT')	3
4. OS DADOS DE SAÍDA ('OUTPUT').....	3
5. O PROCESSAMENTO	4
5.1. CONDUZIR AS OVELHAS AO REDIL.....	4
5.1.1. OS ESTÍMULOS (Ocorrências ou 'Interrupts').....	4
5.1.2. OS MOVIMENTOS	5
5.2. A FUGA DO REDIL	5
5.2.1. MAIS ESTÍMULOS	5
5.2.2. MAIS MOVIMENTOS.....	5
5.3. OS REDIS.....	6
6. PLANEAMENTO	6
6.1. 'COSTUMES'.....	6
6.2. VARIÁVEIS E AVISOS.....	6
7. DESENVOLVIMENTO.....	6
7.1. AS OVELHAS ('costumes').....	6
7.2. OS REDIS ('costumes').....	7
7.3. OS 'SCRIPTS' DOS REDIS.....	7
7.4. OS 'SCRIPTS' DAS OVELHAS.....	8

1. O PROBLEMA

Inicialmente, é indispensável que exista um problema e uma ideia para o solucionar, com a produção dos resultados previstos. Já alguém comparou um programa informático com uma máquina de fazer chouriços em que se introduz um porco de um lado para se obter os enchidos do outro. Assim, de uma maneira simples, pode dizer-se que uma solução informática comporta: dados de entrada, processamento e dados de saída, tal como no esquema seguinte.



No nosso caso, pretendemos encontrar uma maneira aliciante de tratar a equivalência de fracções, provocando os alunos dos 5º e 6º anos do 2º Ciclo e possibilitando-lhes testar os seus conhecimentos nesta matéria. Queremos que o aluno saiba agrupar as fracções conforme as suas equivalências, contando no fim os acertos e os erros cometidos.

2. A IDEIA

Paddle2See, um conceituado ‘scratcher’ de Hampden, Maine, USA, é o autor da seguinte ideia: *“Há um rebanho de ovelhas com fracções escritas no pêlo. Há três ou quatro redis e, de preferência, um cão pastor. O aluno terá que mandar o cão empurrar as ovelhas para os redis, separando-as por grupos de equivalência. Se uma ovelha entrar no redil a que não pertence, faz com que fujam todas as outras já guardadas nesse redil. No fim, somam-se os pontos obtidos pelo aluno”.*

3. OS DADOS DE ENTRADA (‘INPUT’)

Começemos por definir os dados de entrada, coleccionando algumas fracções equivalentes: $\{1/2 ; 2/4 ; 3/6 ; 4/8 ; 5/10\}$ $\{1/3 ; 2/6 ; 3/9 ; 4/12\}$ $\{2/3 ; 4/6 ; 6/9 ; 8/12\}$ $\{1/4 ; 2/8 ; 3/12\}$. Já temos fracções que chegam para um rebanho de 16 ovelhas e 4 redis; mas há grupos de 3, de 4 e de 5 fracções. Se pudermos normalizar a dimensão dos grupos, decerto facilitaremos a programação. Assim, eliminando a 5/10 do primeiro grupo e acrescentando uma 4/16 no último, ficamos com 4 grupos de 4 fracções.

4. OS DADOS DE SAÍDA (‘OUTPUT’)

Como dados de saída ou resultados a obter, pretendemos que as 16 ovelhas sejam “arrumadas” nos 4 redis; e podemos arbitrar 1 ponto por cada ovelha bem arrumada; porém, como queremos que elas fujam quando alguma é mal introduzida, vamos ter de decidir se a penalização dos erros é fixa ou se depende do número de ovelhas que saiam.

Esta segunda opção é atractiva (uma ovelha que entra soma 1 ponto e uma ovelha que sai subtrai 1 ponto); mas não é justa porque o aluno só erra na ovelha mal orientada e não nas certas que, eventualmente, já estariam bem arrumadas.

Por outro lado, se ele errar logo na primeira ovelha, nenhuma sairia e, logo, não haveria penalização. Mas convencionando apenas que uma ovelha certa soma 1 ponto e uma ovelha errada subtrai 1 ponto, aceitamos que a pontuação final de um aluno com erros possa ultrapassar os 16 pontos, já que ele irá “arrumar” várias vezes a mesma ovelha, ao passo que outro aluno que acerte tudo à primeira somará apenas 16 pontos. Isto já está a complicar-se...

O melhor é contabilizar as acções certas e as erradas e apresentar as duas somas no fim sem tecer juízos de valor. O aluno saberá auto-avaliar-se.

5. O PROCESSAMENTO

Falta definir que acções devem ser realizadas para se obter os resultados pretendidos. O ambiente Scratch é, claramente, um ambiente de programação por objectos, ou seja, cada entidade ('sprite') contém em si todas as instruções de que precisa para reagir a estímulos. Assim, é preciso definir que movimentos devem fazer as ovelhas e o cão pastor e também que estímulos devem ser produzidos para que cada um se movimente. Convencionemos desde já que o aluno deverá clicar numa ovelha e no redil para onde ela se deve encaminhar.

5.1. CONDUZIR AS OVELHAS AO REDIL

5.1.1. OS ESTÍMULOS (Ocorrências ou 'Interrupts')

A ovelha que é clicada sabe disso através da instrução 'when sprite clicked' e pode avisar o cão. O mesmo se pode dizer do redil. Vamos numerar os redis e definir uma variável cujo nome "redil activo" já denuncia a sua utilidade; e vamos numerar as ovelhas e definir a variável "ovelha activa" para o mesmo fim.

O aluno clica num redil e ele fica activo. ("redil activo = 3", por exemplo). O aluno clica numa ovelha e ela fica activa. ("ovelha activa = 7", por exemplo). Agora só faltará dar mais um clique no cão para que ele conduza a ovelha nº 7 para o redil nº 3.

É claro que o redil e a ovelha devem mudar de 'costume' para demonstrar que estão activos.

A ovelha tem que calcular a direcção para o redil de destino e o cão tem que calcular a sua direcção para a ovelha activa, rodando em volta dela até se colocar por trás, em relação ao sentido da marcha para o redil. Quando o redil, a ovelha e o cão estiverem colineares, (com a ovelha a meio dos dois), o cão iniciará a sua marcha na direcção da ovelha que, assim, seguirá para o redil activo, já que vamos "ensinar" a ovelha a fugir do cão sempre que este se aproxime a menos de uma dada distância.

Complexo, não é? E falta considerar que a ovelha terá de "sentir" o ângulo de aproximação do cão para se afastar no sentido contrário; e ainda que o movimento do cão fará fugir outras ovelhas que porventura se lhe atravessarem no caminho e que a tendência será para encostar o rebanho aos limites do ecrã, situação em que será difícil ao cão colocar-se por trás das ovelhas.

Chegados aqui, é melhor parar um bocadinho para reavaliar a ideia inicial, porque é preciso ver se a conquista do objectivo justifica a afectação de tantos meios. Também é importante pensar como se poderá explicar o uso de funções trigonométricas aos jovens leitores deste documento, que só agora tomam contacto com os ângulos do plano.

Parece que, tendo como objectivo que os alunos agrupem as fracções segundo a sua equivalência, não se desvirtuará a finalidade do projecto se cada ovelha seguir de "livre vontade" para o redil activo, sem a interferência do cão pastor, mas apenas com o clique do aluno que, aqui, faz a vez da pedrada do pastor. E o programa ficará radicalmente mais simples e sem trigonometria

Acabámos de fazer uma alteração à ideia inicial, depois de já termos gasto tanto tempo a pensar no cão! Mas é mais fácil alterá-la no plano do que, depois, durante a escrita do programa.

Convencionamos então que o aluno deverá clicar numa ovelha para que ela se dirija para o redil previamente activado também com um clique do rato. Estes cliques são o estímulo necessário e suficiente para arrumar cada ovelha; mas temos de prever que, pouco habituado a ler fracções (e muito menos a guardar ovelhas), algum aluno possa aborrecer-se antes de as ter todas aninhadas nos redis, pelo que vamos ter de providenciar uma demonstração de eficácia pastoril, premindo uma tecla qualquer como estímulo para que todas as ovelhas se dirijam mansamente aos seus lares. Este estímulo pode ser a tecla 'space'. Fica combinado!

5.1.2. OS MOVIMENTOS

Os redis costumam ter uma porta por onde passam as ovelhas. Se alargarmos suficientemente essa entrada e colocarmos os redis aos cantos do ecrã, poderemos reduzir o itinerário de cada ovelha a um simples segmento de recta, eliminando o “perigo” de obrigar as ovelhas a saltarem a vedação. Ainda bem que só temos quatro redis, pois tantos são os cantos do ecrã. Com mais redis, os movimentos iriam complicar-se.

Se marcarmos quatro posições distintas, dentro de cada redil, evitamos que as ovelhas se atropelem e fiquem umas em cima das outras. Então, quando uma ovelha se dirigir ao seu redil, sabe exactamente onde deve deitar-se. Se não tem ali lugar, não é o seu redil e, logo, deve fugir, arrastando consigo todas as que, porventura, já ali estejam acomodadas.

Este movimento de recolha deverá ser executado com uma instrução “glide n secs X: Y:” para que seja visível o movimento da ovelha. A fuga será tratada já a seguir.

5.2. A FUGA DO REDIL

5.2.1. MAIS ESTÍMULOS

Quando uma ovelha chega ao redil activo e verifica que não é o seu, deve deslizar para a sua posição inicial e emitir um aviso para aquelas que eventualmente já estejam nesse redil a acompanharem. Na verdade, antes de se deslocar para o mau redil, a ovelha já pode saber que não é o seu, mas deve fazer, na mesma, o movimento, para se perceber que ela ia errada e para se ver que as outras saem atrás dela.

Assim, não precisa de estímulo para fugir dali; mas tem que estimular as outras a seguirem-na e o meio mais indicado é com uma instrução “broadcast aviso”.

5.2.2. MAIS MOVIMENTOS

Isto implica que todas as ovelhas possuam instruções para se deslocarem aos seus pontos iniciais quando recebem o aviso para saírem do seu redil. (Mesmo que ainda lá não estejam, não há problema, porque se esforçam para ficarem, afinal, no lugar onde estão).

Seria mais interessante se, em vez de voltarem aos seus lugares iniciais, as ovelhas fugissem aleatoriamente, para que o aluno não soubesse já onde ir buscá-las a segunda vez. O esforço de programação é o mesmo: em vez de usar coordenadas na instrução ‘glide’ pode usar-se a

função 'random', mas enfatizar o erro não é boa prática. Assim, convencionamos que as ovelhas fugitivas se devem dirigir aos lugares que ocupam quando se clica na bandeira verde.

5.3. OS REDIS

Há quatro redis, um a cada canto do ecrã, e há sempre um deles activo porque o aluno pode logo começar por clicar uma ovelha que ficaria muito embaraçada se não houvesse um redil activo. Depois, quando um dos redis recebe um clique para se activar, deve mudar de 'costume' e avisar os seus congéneres para se desactivarem, esteja ou não esteja um deles activo. (É que o aluno pode até activar o que já está activo, e o efeito será sempre o de se activar um e desactivar três dos redis).

6. PLANEAMENTO

6.1. 'COSTUMES'

Ficando aos cantos, os redis devem ter a forma de sector circular com 90º de amplitude para se adaptarem bem ao lugar. Internamente, devem ter marcadas as posições finais das ovelhas, com as cabeças viradas para a gamela da ração, por forma a que se distingam bem os lugares vazios. Isto implica que, depois de chegarem ao lugar, as ovelhas devem rodar as cabeças na direcção do vértice do sector circular. Cada redil terá dois 'costumes': um para a situação de redil activo e outro para quando está desactivado.

As ovelhas também precisam de dois 'costumes' para se distinguir bem a que se movimenta e adormece no redil, ou seja, a ovelha activa, segundo a terminologia que temos vindo a usar. No dorso das ovelhas serão inscritas as fracções já definidas em **3**.

6.2. VARIÁVEIS E AVISOS

Pode haver necessidade de definir mais variáveis mas, pelo menos, haverá uma ovelha activa, uma redil activo, uma Certas e uma Erradas, estas para acumularem os resultados do aluno. Têm que existir quatro avisos de fuga, um para cada grupo de ovelhas e ainda um outro para informar que um redil foi activado. (Em tempo: ovelha activa foi apagada por não ser necessária)

7. DESENVOLVIMENTO

Ainda não escrevemos uma única instrução, ainda não deixámos de falar em generalidades, ainda não saímos das conjecturas e já lá vão quatro páginas de texto. Mas não foi tempo perdido: há imensas variantes de que não nos ocuparemos e já temos uma ideia bastante aproximada do que pretendemos fazer. É tempo de passarmos ao computador para testar, por exemplo, se estes movimentos a que nos temos vindo a referir funcionam tal como imaginámos.

7.1. AS OVELHAS ('costumes')

Esta ovelha estilizada pode servir perfeitamente ao fim em vista; mas, se encontrarmos outra imagem mais sugestiva, também não será difícil importá-la para substituir esta. Desde já, ressalta um aspecto importante que não foi previsto: o texto das fracções fica ilegível quando as ovelhas rodam. Assim,



Fig 1

caem por terra as ideias de as fazer caminhar com a cabeça para a frente e ficar na gamela viradas para o vértice do redil. (E nos outros quadrantes, os textos até ficariam invertidos).

A cor branca pode ser substituída por um cinzento para indicar que a ovelha está activa. Logo, ficamos com dois ‘costumes’ a que chamaremos “Activa” e “Inactiva”. O tamanho das ovelhas que pareceu mais adequado à legibilidade das fracções, ao espaço do ecrã, (que será o prado) e ao tamanho dos redis foi de aproximadamente 35x15 ‘pixels’ (‘pixel’=dimensão de um ponto).

7.2. OS REDIS (‘costumes’)

Esta imagem de um redil não é a que primeiro foi desenhada. Na primeira, de acordo com o plano, o lugar das ovelhas tinham as cabeças viradas para o vértice. Depois, como ficou decidido que as ovelhas não rodavam, foi preciso refazer este redil, para ficar em conformidade. Ainda bem que só havíamos desenhado um redil, senão teríamos quatro para emendar. É um bom princípio não copiar logo os primeiros ‘sprites’, porque copiar é fácil, emendar erros multiplicados é que se torna difícil.

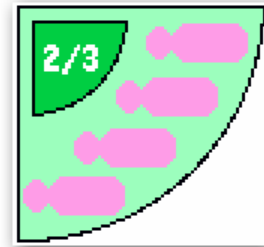


Fig 2

Para sinalizar o redil activo, faremos outro ‘costume’ com a gamela vermelha e chamaremos “Inactivo” a um e “Activo” a outro. Um bom tamanho para os redis pode ser Raio = 80 ‘pixels’.

7.3. OS ‘SCRIPTS’ DOS REDIS

Começamos por colocar os redis nos cantos do ecrã, levando-os para lá como rato e, depois, trazendo de ‘Motion’ uma instrução “go to X: Y.” que, como se sabe, é actualizada para a posição que o ‘sprite’ tiver no ecrã de desenho. Quando a bandeira verde for clicada, cada redil posiciona-se no seu lugar com o ‘costume’ “Inativo”.

Quando o redil for clicado, deve mudar para o ‘costume’ “Activo”, ao mesmo tempo que actualiza a variável Redil Activo que é aquela que guarda o número do redil activo; também é preciso avisar os outros redis para ficarem inactivos. Assim, é preciso uma instrução “broadcast RedilActivo” no mesmo bloco “when Redil clicked”.



Fig 3

Claro que vamos precisar de uma “when I receive RedilActivo” para que este redil mude para o ‘costume’ “Inactivo” quando outro redil ficar activo e enviar o aviso; mas neste caso, assim que fosse emitido esse aviso, todos os redis ficariam inactivos, incluindo o redil emissor do aviso. Isso leva-nos a criar uma condição no bloco “when I receive” em que, ao receber o aviso, cada redil verifica, através do valor da variável Redil Activo que ‘costume’ deve envergar. Neste caso, já não é preciso ficarmos com a instrução “switch to costume Activo” no bloco “when Redil clicked”. E fica garantido que apenas um redil está activo num dado momento.

Falta garantir que, desde o início, há um redil activo, mesmo antes do aluno clicar num deles. É o “valor por defeito” que deve ser assegurado num dos redis, acrescentando no bloco “when bandeira clicked” estas duas instruções em lugar da “switch to costume Inactivo”, como a figura 4 ilustra.



Fig 4

Agora, convém copiar o redil e testar se ambos se comportam como o previsto. É claro que é preciso corrigir os valores numéricos do novo redil, para o posicionar no seu lugar e para que não fiquem os dois a activar-se e desactivar-se, ao mesmo tempo em vez de se alternarem. Quando estivermos satisfeitos com o comportamento das ovelhas, logo copiaremos estes redis para os que ainda faltam, adequando-lhes também as coordenadas e o valor de Redil Activo.

7.4. OS ‘SCRIPTS’ DAS OVELHAS

Como temos de colocar cada ovelha num dado ponto do prado, logo que a bandeira é clicada; e como temos que a conduzir ao redil, calmamente, quando é premida a ‘space’, vamos criar esses dois blocos, afinando as coordenadas e testando se a ovelha se comporta como esperado. Podemos planejar desde já, num papel à parte, a ocupação do espaço do ecrã pelas ovelhas, numerando-as e atribuindo-lhes redil e fracção.



Fig 5

Convém copiar esta ovelha para criar outra do outro redil, para testarmos como respondem a estes dois estímulos do clique na bandeira verde e da acção sobre a tecla ‘space’.

O passo seguinte é o de programar o movimento da ovelha quando é clicada. Embora o bloco final seja o apresentado na figura 6, vamos começar por construir três instruções “if...else” aninhadas para que as diferentes instruções sejam cumpridas conforme o redil que estiver activo e nunca no caso contrário.

Repare-se na lógica do bloco da figura 6: Se Redil Activo = 1 cumprem-se as instruções de que falaremos a seguir; senão, testamos se Redil Activo = 2 para cumprir outras instruções; se ainda não for esse o caso, tornamos a testar se Redil Activo=3 e, se também não for, então já se sabe que Redil Activo = 4.



Fig 6

Admitindo que esta ovelha pertence ao 1º redil e que este está activo, vamos fazê-la deslizar para o seu lugar dentro dele, com as coordenadas usadas no bloco “when space key pressed”; e vamos incrementar a variável Certas pois o aluno acertou esta equivalência. Adiante veremos como foi adicionada a instrução do som.

Mas se o redil activo não for o 1º mas antes o 2º (a que a ovelha não pertence), ela deve seguir para as coordenadas do 2º e, depois de lançar o aviso “Fugam02” (que pretende ser ouvido por

todas as ovelhas que pertençam ao 2º redil), regressa à sua posição de início, (já usada no bloco “when bandeira clicked”), onde fica inactiva.

O mesmo se passará em relação aos restantes redil, que nesta altura ainda não foram criados. A verdade é que estas duas instruções de voltar à posição inicial e ficar inactiva é comum aos casos em que o redil activo não é o primeiro. Assim, estas instruções e a contagem das Erradas devem estar fora das “if...else” dos três redil errados, mantendo-se dentro do “else” ao 1º redil.

Agora, podemos criar, por cópia, mais três ovelhas para encher o primeiro redil, adequando as coordenadas porque, no resto, os ‘scripts’ destas ovelhas são iguais aos da sua companheira do 1º redil. É preciso ser muito cuidadoso nestas cópias porque se erra muito frequentemente, por omissão, ao deixar nas cópias certas características exclusivas do modelo copiado.

Como temos uma ovelha que não pertence a este redil, ficamos habilitados a testar o que vai acontecer quando esta intrusa se dirigir ao primeiro redil. Vamos copiar o bloco da nossa primeira ovelha para esta, adequando o “if...else” como se vê na figura 7 em que se trocaram as posições do primeiro e do segundo redil, acertando-se, é claro, as coordenadas.



Fig 7

Repare-se que o aviso lançado por esta ovelha, quando vai ao primeiro redil (que não é o seu), passou agora a ser “Fujam01”.

Só falta dizer às quatro ovelhas do primeiro redil o que devem fazer quando recebem o aviso “Fujam01”. Por isso, é preciso acrescentar em cada uma delas o bloco desta figura 8, em que a ovelha muda o ‘costume’ para “Inactiva” e desliza para a sua

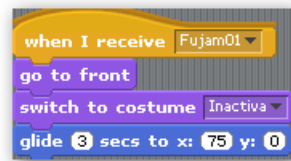


Fig 8

posição inicial, (cujas coordenadas já conhecemos). A outra instrução “go to front” destina-se a evitar que a ovelha que se desloca passe por baixo das que estão paradas e, por isso é que existe outra instrução destas no bloco “when Ovelha clicked” (e só não existe no bloco “when space key pressed” porque o problema não existe quando todas se movem ao mesmo tempo).

Depois de termos testado o comportamento destas cinco ovelhas, é tempo de se criarem (por cópia), os dois redil e as onze ovelhas que faltam, tendo o cuidado de acertar os pormenores que são exclusivos de cada um. Foi isso que fizemos, antes de nos lembrarmos que ficaria bem acrescentar um qualquer som a sinalizar as ovelhas bem direccionadas e outro quando as ovelhas fugissem do redil. Por isso, ficou mais difícil acrescentar em cada ovelha a instrução “play sound Laser2” e em quatro delas a “play sound Fairydust”.

Não foi ainda mencionado, mas o ‘stage’ foi colorido com um tom de verde e salpicado com umas ervas para dar a ideia de um prado. As variáveis Certas e Erradas foram assinaladas para aparecerem no ecrã, onde também se incluiu o símbolo dos autores da ideia e do projecto

Março de 2008

Fernando Frederico
efe.fred@gmail.com