






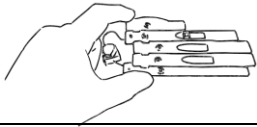
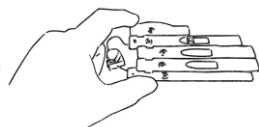


Liste des blocs Scratch reconnus

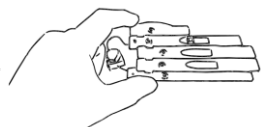
# ID	Signification du bloc	topcode		opcode	[inputs]			
1	Quand drapeau cliqué	31		event_whenflagclicked				
2	Quand ... est pressé	47		event_whenkeypressed	F	KEY_OPTION	space	
3	Si ...alors	55		control_if	CONDITION	SUBSTACK		
4	Répéter Jusqu'à ce que	59		control_repeat_until	CONDITION	SUBSTACK		
5	Répéter ... fois	61		control_repeat	TIMES	SUBSTACK		
6	Répéter indéfiniment	79		control_forever	SUBSTACK			
7	Attendre ... secondes	87		control_wait	DURATION			











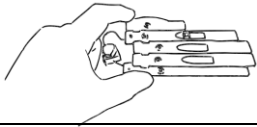
8	Sinon*	91		control_if_else	CONDITION	SUBSTACK	SUBSTACK 2	
9	Fin de boucle/ Fin Si	93		control_fin				
10	Stop tout	103		control_stop	F	STOP_OPTIO N	all	
11	Montrer	107		looks_show				
12	Cacher	109		looks_hide				
13	Dire ... pendant ... secondes	115		looks_sayforsecs	MESSAGE	SECS		
14	Dire	117		looks_say	MESSAGE			





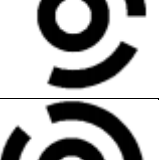





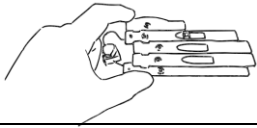
15	Effacer tout	121		pen_clear				
16	Stylo en position basse (position d'écriture)	143		pen_penDown				
17	Stylo en position haute (relever le stylo)	151		pen_penUp				
18	Avancer de ...	155		motion_movesteps	STEPS			
19	Tourner de ... (sens horaire)	157		motion_turnright	DEGREES			
20	Tourner de ... (sens anti-horaire)	167		motion_turnleft	DEGREES			
21	S'orienter à ...	171		motion_pointindirection	DIRECTION			
22	Aller à x : ... y : ...	173		motion_gotoxy	X	Y		



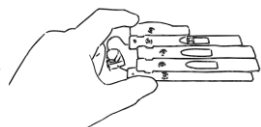
23	Ajouter ... à x	179		motion_changexby	DX			
24	Mettre x à :	181		motion_setx	X			
25	Ajouter ... à y	185		motion_changeyby	DY			
26	Mettre y à :	199		motion_sety	Y			
27	Abscisse x	203		motion_xposition				
28	Abscisse y	205		motion_yposition				
29	Direction	211		motion_direction				
30	Définir Bloc 1	213		procedures_definition				



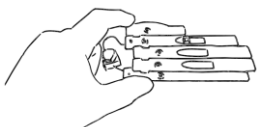
31	Définir Bloc 2	217		procedures_definition				
32	Définir Bloc 3	227		procedures_definition				
33	Bloc 1	229		procedures_call				
34	Bloc 2	233		procedures_call				
35	Bloc 3	241		procedures_call				
36	Réponse	271		sensing_answer				
37	Touche ... pressée ?	279		sensing_keypressed	F	KEY_OPTION	space	
38	Demander ... et attendre	283		sensing_askandwait	QUESTION			





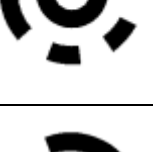



39	... + ...	285		operator_add	NUM1	NUM2		
40	... - ...	295		operation_subtract	NUM1	NUM2		
41	... * ...	299		operator_multiply	NUM1	NUM2		
42	... / ...	301		operator_divide	NUM1	NUM2		
43	... < ...	307		operator_lt	OPERAND1	OPERAND2		
44	... > ...	309		operator_gt	OPERAND1	OPERAND2		
45	... = ...	313		operator_equals	OPERAND1	OPERAND2		
46	... et ...	327		operator_and	OPERAND1	OPERAND2		



47	... ou ...	331		operator_or	OPERAND1	OPERAND2		
48	... non ...	333		operator_not	OPERAND			
49	Nombre aléatoire entre ... et ...	339		operator_random	NUM1	NUM2		
50	Regroupe ... et ...	341		operator_join	STRING1	STRING2		
51	Modulo ...	345		operator_mod	NUM1	NUM2		
52	Arrondi de ...	355		operator_round	NUM			
53	Mettre ... à ...	369		data_setvariableto	VALUE	F	VARIABLE	VARIABLE
54	Ajouter ... à ...	391		data_changevariableby	VALUE	F	VARIABLE	VARIABLE



55	Montrer la variable ...	395		data_showvariable	F	VARIABLE	VARIABLE	
56	Cacher la variable ...	397		data_hidevariable	F	VARIABLE	VARIABLE	
57	Variable quart_de_tour **	403		data_quart				
58	Variable demi_tour ***	405		data_demi				
59	Variable var1	357		data_var1				
60	Variable var2	361		data_var2				
61	Variable chat_parle	425		data_chatparle				

62	Cacher la liste	457		data_hidelist	F	LIST	LIST	
63	Montrer la liste	465		data_showlist	F	LIST	LIST	
64	Supprimer toute la liste	551		data_deletealloflist	F	LIST	LIST	
65	Élément de la liste	555		data_itemoflist	F	LIST	LIST	
66	Ajoute à la liste	563		data_addtolist	ITEM	F	LIST	LIST

* Sinon => se rattache au premier « Si ... Alors ... » non finit et le transforme en « Si ... Alors ... Sinon ... »

** Valeur prédéfinie : quart_de_tour = 90

*** Valeur prédéfinie : demi_tour = 180

À noter que pour le feedback, le programme doit terminer, c'est-à-dire pas de boucle infini : répéter indéfiniment ou autre

Annexe - Structure d'un bloc Scratch

Un programme Scratch (.sb3) est en fait un dossier zippé contenant un fichier au format JSON et des ressources

```
"ID DU BLOC" :
{
  "opcode": "OPCODE DU BLOC",
  "next": "ID DU BLOC SUIVANT CE BLOC",
  "parent": "ID DU BLOC PARENT DE CE BLOC",
  "inputs": {LISTE DES INPUTS DU BLOC},
  "fields": {LISTE DES FIELDS DU BLOC},
  "shadow": BOOLEAN,
```



```
"topLevel": BOOLEAN,
"x": ABS,
"y": ORD}
```

A propos des formes de blocs

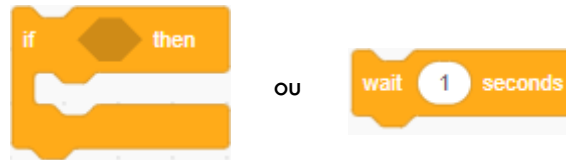
Les formes de chaque bloc sont disponibles ici : https://en.scratch-wiki.info/wiki/Blocks#Block_Shapes

Remplir la partie inputs

La partie **inputs** est une liste construite entre des accolades dont les champs sont séparés par des virgules. {Champ1, Champ2, ..., ChampN}

Les **inputs** d'un bloc sont des entrées qui ne sont pas dans un menu déroulant dans Scratch.

Prenons par exemple les blocs suivants :



Ces deux blocs montrent les principaux types d'inputs qui existent. Dans le cas du bloc « **if** » on ne peut qu'entrer un bloc (que cela soit en condition ou dans le corps de la partie après le « **then** »), on ne peut ni taper une entrée ni y mettre une variable.

Dans le cas du bloc « **wait** » on peut mettre y mettre un bloc, y entrer des données directement ou y mettre une variable.

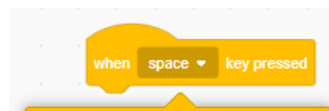
Voici comment construire chacun de ces types d'inputs :

- Dans le cas d'un input ne prenant en entrée qu'un bloc (comme le « **if** »), le champ à entrer dans la liste sera :
"NOM DU CHAMP" : [TYPE DE L'ENTRÉE, ID DU BLOC]
- Dans le cas d'un input pouvant tout prendre en entrée (comme « **wait** ») le champ à entrer dans la liste sera :
"NOM DU CHAMP" : [SHADOW, VALUE]
 - avec **SHADOW** = 1 dans le cas d'une entrée directe
 - et **SHADOW** = 3 dans le cas d'un bloc ou d'une variable
 - et **VALUE** = [TYPE DE L'ENTRÉE, "ENTRÉE"] dans le cas d'une entrée directe
 - "ID DU BLOC", [TYPE DE L'ENTRÉE, "ENTRÉE PAR DÉFAUT"] dans le cas d'un bloc
 - [12, "NOM DE LA VARIABLE", "ID DE LA VARIABLE"], [TYPE DE L'ENTRÉE, "ENTRÉE PAR DÉFAUT"] dans le cas d'une variable

Remplir la partie fields

La partie **fields** est une liste construite entre des accolades dont les champs sont séparés par des virgules. {Champ1, Champ2, ..., ChampN}

Les **fields** d'un bloc sont des entrées qui se trouvent dans un menu déroulant sur le bloc Scratch.



Prenons par exemple les blocs :

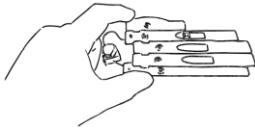
:ou



Ces deux blocs montrent les deux types de **fields** qui existent, le premier a une liste définie et immuable d'options dans lesquelles choisir alors que le deuxième a une liste qui dépend de l'état actuel de notre environnement Scratch.

Voici comment construire un champ de la liste des **fields** d'un bloc :

- Dans le cas du premier type de blocs le champ à entrer sera : "NOM DU CHAMP": [OPTION, null]
- Dans le cas du deuxième type de blocs, les seuls que nous avons dans la liste demandée sont ceux qui concernent les variables (à voir s'il y en a d'autres). Pour ces blocs la syntaxe sera : `VARIABLE: [nom de la variable, ID de la variable]`



Cas d'un bloc personnalisé

La création d'un bloc personnalisé ressemble à celle de n'importe quel bloc et a un champ d'input qui a pour syntaxe "custom_block": [1, "ID DU BLOC"] avec ID DU BLOC = l'ID qu'on utilisera pour créer le prototype de ce bloc et le champ **next** sera le premier bloc du corps de ce bloc personnalisé. (l'opcode à utiliser est « **procedures_definition** »).

Quand on crée un bloc personnalisé un bloc prototype est donc créé, ce bloc a pour opcode « **procedures_prototype** » et a une syntaxe un peu différente des autres blocs :

```
"ID DU BLOC" :
  {"opcode": "procedures_prototype",
   "next": "ID DU BLOC SUIVANT CE BLOC",
   "parent": "ID DU BLOC PARENT DE CE BLOC",
   "inputs": {},
   "fields": {},
   "shadow": BOOLEAN,
   "topLevel": BOOLEAN,
   "x": ABS,
   "y": ORD}
  "mutation":
    {"tagName": "mutation",
     "children": [],
     "proccode": "NOM DU BLOC",
     "argumentids": "[]",
     "argumentnames": "[]",
     "argumentdefaults": "[]",
     "warp": "BOOLEAN"}
```

L'utilisation d'un bloc personnalisé a aussi le champ **mutation** à remplir pour l'utiliser.

```
"ID DU BLOC" :
  {"opcode": "procedures_call",
   "next": "ID DU BLOC SUIVANT CE BLOC",
   "parent": "ID DU BLOC PARENT DE CE BLOC",
   "inputs": {},
   "fields"
   " shadow" BOOLEAN,
   "topLevel": BOOLEAN,
   "x": ABS,
   "y": ORD}
```