

MP-Scratch

METAPOST au service de l'algorithmique

CHRISTOPHE POULAIN

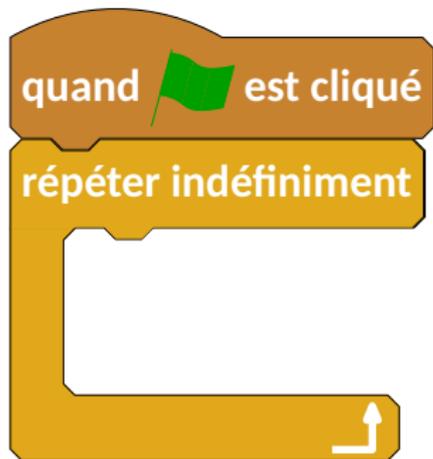
christophe.poulain@melusine.eu.org

Collège Paul Eluard

Un exemple



Un exemple



Un exemple

quand  est cliqué

répéter indéfiniment

dire

Ce stage est super ! Vive METAPOST ! $\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$



Un exemple



```
input mp-scratch
```

```
beginfig (1);
draw Drapeau;
draw RepeterI;
draw Dire (" \opSimple{Ce_stage_est_super !_Vive _\MP !_\$ \displaystyle \sum_{n=1}^{\infty}
\frac{1}{n^2}=\frac{\pi^2}{6}}");
draw FinBlocRepeterI;
endfig;
end;
```

Algorithme en \LaTeX

- package **listings**

Algorithme en \LaTeX

- package **listings**

```
Demander un nombre  
Ajouter 2  
Multiplier par 3  
Retrancher 2  
Afficher le r\`esultat
```

Algorithme en \LaTeX

- package **listings**
- package **algorithms**

Algorithme en \LaTeX

- package **listings**
- package **algorithms**

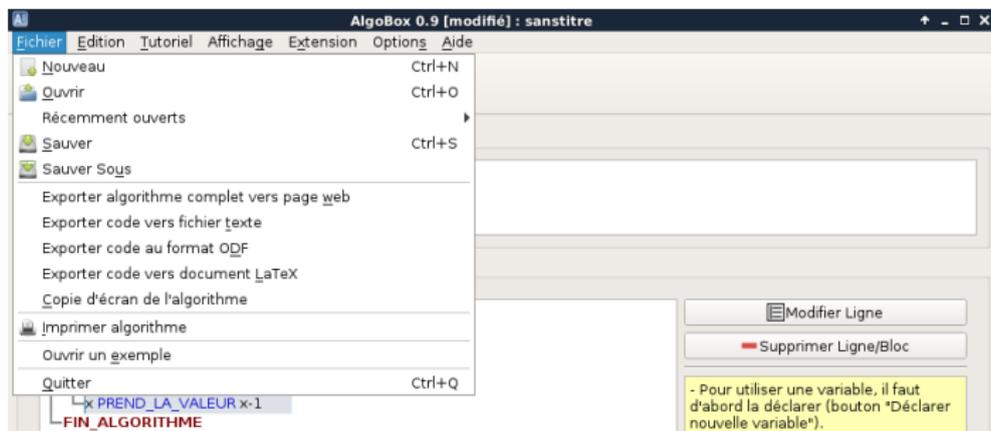
-
- 1: Demander un nombre n
 - 2: $n + 2 \rightarrow p$
 - 3: $p * 3 \rightarrow q$
 - 4: $q - 2 \rightarrow r$
 - 5: Afficher le résultat r
-

Algorithmes en \LaTeX

- package **listings**
- package **algorithms**
- **algobox**

Algorithme en \LaTeX

- package **listings**
- package **algorithms**
- **algotbox**



Algorithme en \LaTeX

- package **listings**
- package **algorithms**
- **algotext**

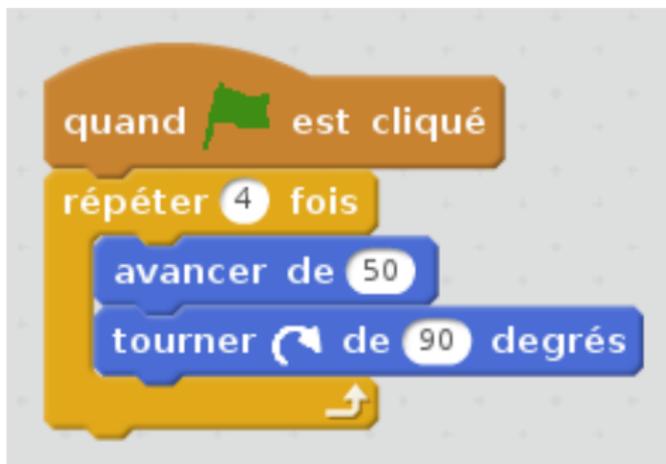
```
1: VARIABLES
2: x EST_DU_TYPE NOMBRE
3: DEBUT_ALGORITHME
4:   x PREND_LA_VALEUR x+1
5:   x PREND_LA_VALEUR 3x
6:   x PREND_LA_VALEUR x-1
7: FIN_ALGORITHME
```

Algorithme en \LaTeX

- package **listings**
- package **algorithms**
- **algotext**
- captures d'écran

Algorithme en \LaTeX

- package **listings**
- package **algorithms**
- **algotext**
- captures d'écran



Pourquoi MP-Scratch ?

Pourquoi MP-Scratch ?

- Pourquoi pas ? :)

Pourquoi MP-Scratch ?

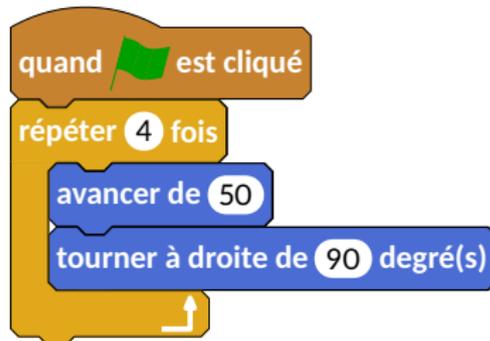
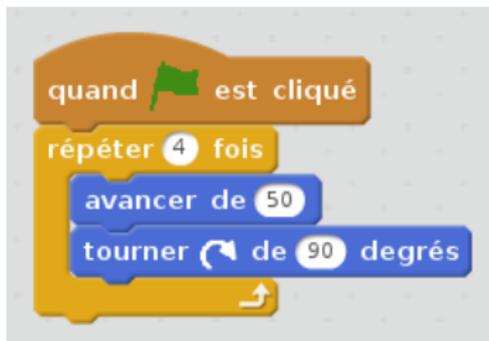
- Pourquoi pas ? :)
- Basé sur METAPOST

Pourquoi MP-Scratch ?

- Pourquoi pas ? :)
- Basé sur METAPOST
- Introduction de l'algorithmique au collège.

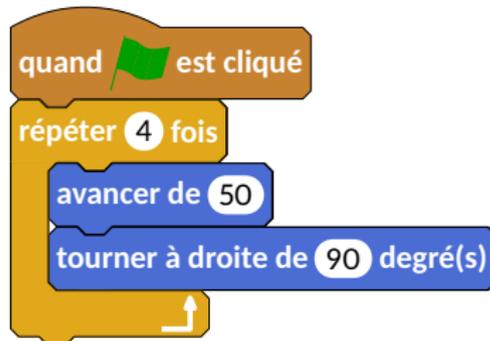
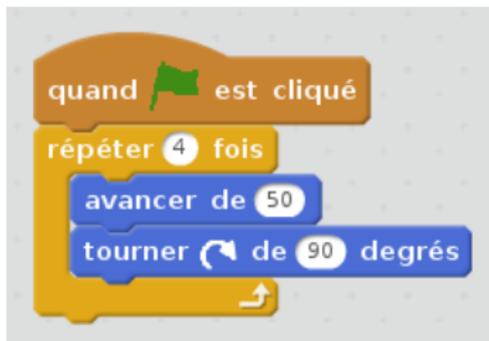
Pourquoi MP-Scratch ?

- Pourquoi pas ? :)
- Basé sur METAPOST
- Introduction de l'algorithmique au collège.
- Capture d'écran



Pourquoi MP-Scratch ?

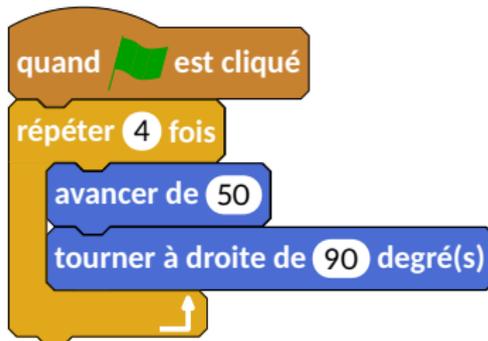
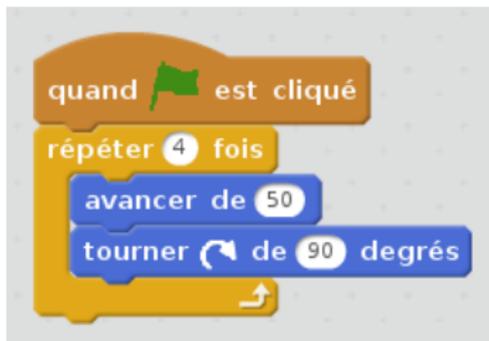
- Pourquoi pas ? :)
- Basé sur METAPOST
- Introduction de l'algorithmique au collège.
- Capture d'écran



- Typographie, qualité et homogénéité des documents.

Pourquoi MP-Scratch ?

- Pourquoi pas ? :)
- Basé sur METAPOST
- Introduction de l'algorithmique au collège.
- Capture d'écran



- Typographie, qualité et homogénéité des documents.
- Algorithmique débranché

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :
 - scratch : <https://ctan.org/pkg/scratch>

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :
 - scratch : <https://ctan.org/pkg/scratch>
 - scratchx : <https://www.ctan.org/pkg/scratchx>

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :
 - scratch : <https://ctan.org/pkg/scratch>
 - scratchx : <https://www.ctan.org/pkg/scratchx>
 - tikzcodeblocks :
<https://www.ctan.org/pkg/tikzcodeblocks>

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :
 - scratch : <https://ctan.org/pkg/scratch>
 - scratchx : <https://www.ctan.org/pkg/scratchx>
 - tikzcodeblocks :
<https://www.ctan.org/pkg/tikzcodeblocks>
- basé sur pstricks :

Autres packages disponibles sous L^AT_EX

- basé sur TikZ :
 - scratch : <https://ctan.org/pkg/scratch>
 - scratchx : <https://www.ctan.org/pkg/scratchx>
 - tikzcodeblocks :
<https://www.ctan.org/pkg/tikzcodeblocks>
- basé sur pstricks :
 - scratchTeX :
<https://github.com/nicolaspoulain/scratchTeX>

Installation

- Récupérer l'archive.

`http://melusine.eu.org/syracuse/G/mp-scratch/`

Installation

- Récupérer l'archive.

`http://melusine.eu.org/syracuse/G/mp-scratch/`

- Vérification de la disponibilité des packages particuliers.

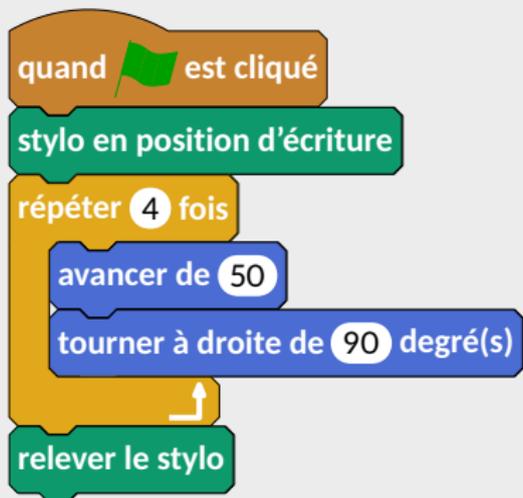
Installation

- Récupérer l'archive.

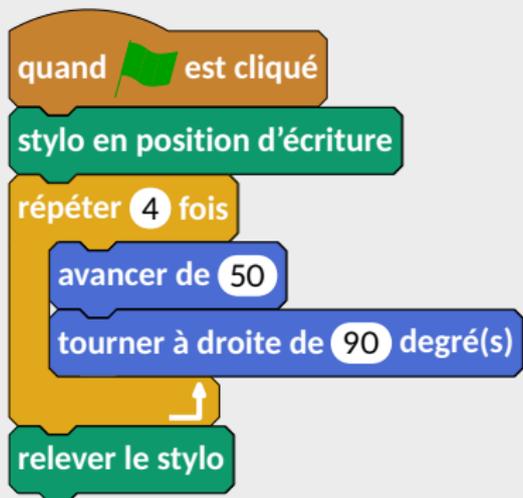
`http://melusine.eu.org/syracuse/G/mp-scratch/`

- Vérification de la disponibilité des packages particuliers.
- Installation dans *un répertoire local*.

À vous de jouer

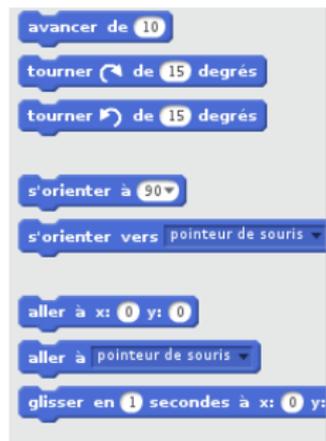


À vous de jouer

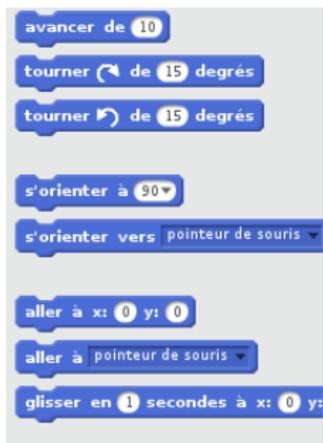


```
draw Drapeau ;  
draw PoserStylo ;  
draw Repeter ("4") ;  
draw Avancer ("50") ;  
draw Tournerd ("90") ;  
draw FinBlocRepeter ;  
draw ReleverStylo ;
```

Groupe « Mouvement »



Groupe « Mouvement »



- **draw** Avancer("10");
- **draw** Tournerd("90");
- **draw** Tournerg("90");
- **draw** Orienter("90");

avancer de 10

tourner à droite de 15 degré(s)

tourner à gauche de 15 degré(s)

s'orienter à 90

- **draw** Orienterdirection("pointeur_de_souris");

s'orienter vers pointeur de souris

- **draw** Aller("50","100");

aller à x: 50 y: 100

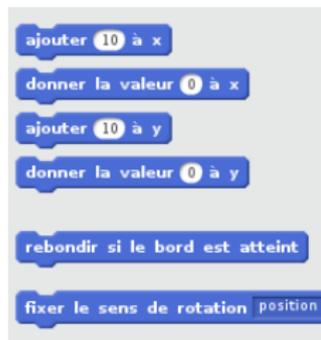
- **draw** Allera("pointeur_de_souris");

aller à pointeur de souris

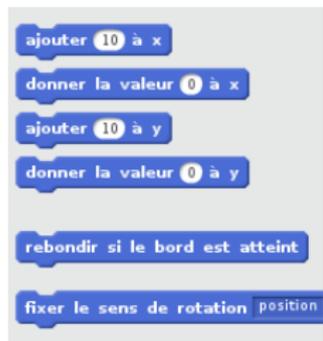
- **draw** Glisser("2","50","100");

glisser en 2 seconde(s) à x: 50 y: 100

Groupe « Mouvement »



Groupe « Mouvement »



- **draw** Ajouter("10","x");
- **draw** Mettre("10","x");
- **draw** Ajouter("50","y");
- **draw** Mettre("10","y");
- **draw** Rebondir;

ajouter 10 à x

donner la valeur 10 à x

ajouter 50 à y

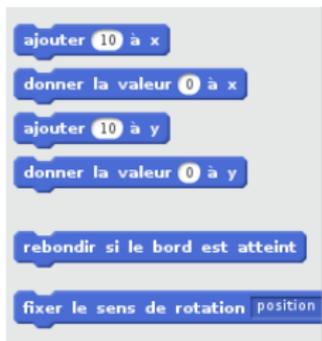
donner la valeur 10 à y

rebondir si le bord est atteint

- **draw** FixerSensRotation("position_\'a_gauche_ou_\'a_droite");

fixer le sens de la rotation position à gauche ou à droite ▼

Groupe « Mouvement »



- **draw** Ajouter("10","x");
- **draw** Mettre("10","x");
- **draw** Ajouter("50","y");
- **draw** Mettre("10","y");
- **draw** Rebondir;

ajouter 10 à x

donner la valeur 10 à x

ajouter 50 à y

donner la valeur 10 à y

rebondir si le bord est atteint

- **draw** FixerSensRotation("position_\'a_gauche_ou_\'a_droite");

fixer le sens de la rotation position à gauche ou à droite ▼

- **draw** Avancer("\opOp{\$\opMouv{abscisse_x}\bm{+}\opSimple{10}\$)");

avancer de abscisse x + 10

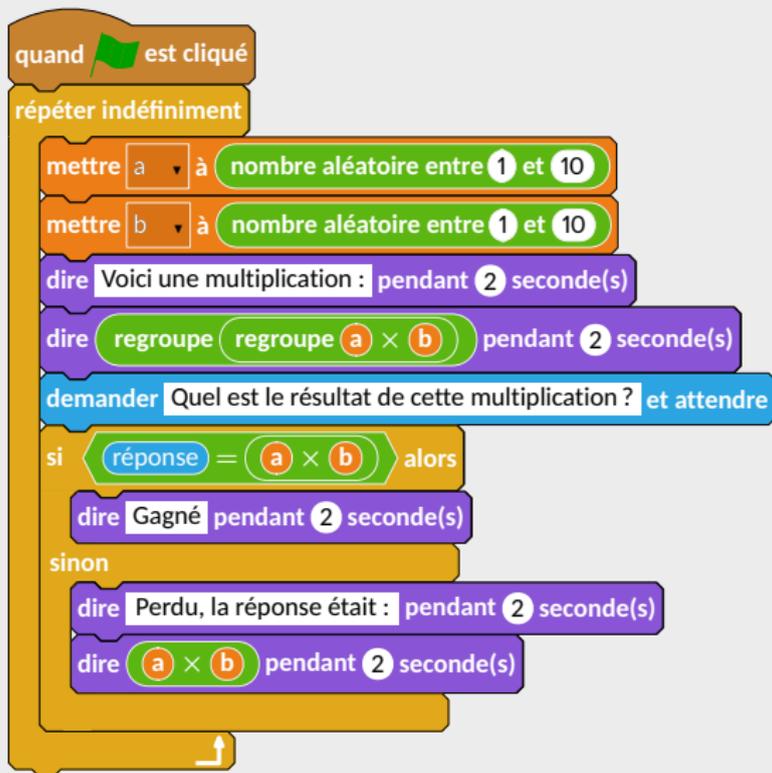
- **draw** Mettre("\opMouv{abscisse_x}","y");

donner la valeur abscisse x à y

- **draw** Ajouter("\opMouv{direction}","y");

ajouter direction à y

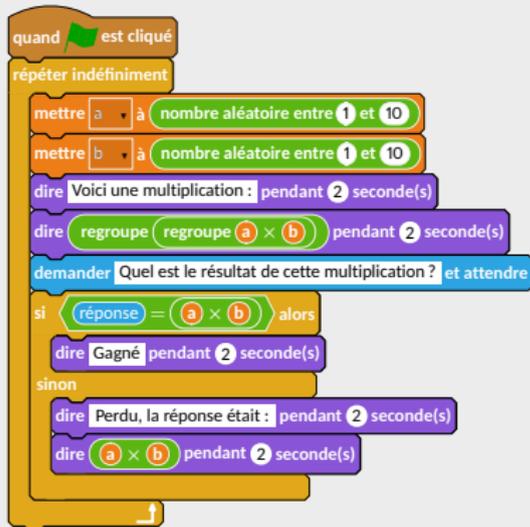
À vous de jouer !



```

draw Drapeau;
draw RepeterI;
picture BB[];
BB1=OvalOp("nombre aléatoire entre ",OvalNb("1")," et
",OvalNb("10"));
draw MettreVar("a",BB1);
draw MettreVar("b",BB1);
draw DireT("Voici une multiplication :", "2");
BB2=OvalOp("regroupe",OvalVar("a"),"
$\\bm{\\times}$",OvalVar("b"));
draw DireT(OvalOp("regroupe",BB2),"2");
draw Demander("Quel est le résultat de cette
multiplication?");
BB3=OvalOp(OvalVar("a")," $\\bm{\\times}$"
",OvalVar("b"));
BB4=TestOp(OvalCap("réponse"),"$\\bm{=}$",BB3);
draw Si(BB4);
draw DireT("Gagné","2");
draw Sinon;
draw DireT("Perdu, la réponse était :", "2");
draw DireT("Perdu, la réponse était :", "2");
draw DireT(BB3,"2");
draw FinBlocSi;
draw FinBlocRepeter;

```



Groupe « Apparence »



Groupe « Apparence »



- **draw** Basculer("costume2");
- **draw** CostumeSuivant;
- **draw** BasculerAR("arriere-plan2");

- **draw** DireT("Hello",2);

dire Hello pendant 2 seconde(s)

- **draw** Dire("Hello");

dire Hello

- **draw** PenserT("Hmm...", "2");

penser Hmm... pendant 2 seconde(s)

- **draw** Penser("Hmm...");

penser Hmm...

- **draw** Montrer;

montrer

- **draw** Cacher;

cachez

basculer sur le costume costume2

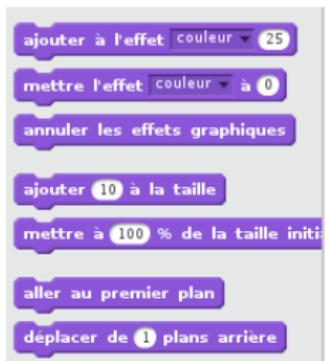
costume suivant

basculer sur l'arrière-plan arrière-plan2

Groupe « Apparence »



Groupe « Apparence »



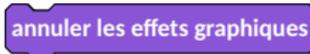
- **draw** AjouterEffet("couleur", "10");



- **draw** MettreEffet("couleur", "10");



- **draw** AnnulerEffet;



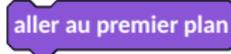
- **draw** AjouterTaille("10");



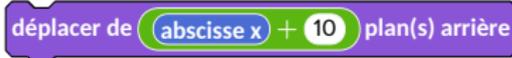
- **draw** MettreA(OvalOp(OvalNb("10"), "\$\bm{+}\$", OvalNb("5")));



- **draw** AllerPPlan;



- **draw** DéplacerAP(OvalOp(OvalMouv("abscisse_x"), "_\$\bm{+}\$_", OvalNb("10")));



Groupe « Apparence »

Quand « la scène » est sélectionnée, on dispose également des commandes :

- **draw** BasculerARA("arriere-plan2");

basculer sur l'arrière-plan et attendre

- **draw** ARSuivant;

arrière-plan suivant

Groupe « Apparence »

Quand « la scène » est sélectionnée, on dispose également des commandes :

- **draw** BasculerARA("arriere-plan2");

basculer sur l'arrière-plan arriere-plan2 et attendre

- **draw** ARSuivant;

arrière-plan suivant

Les « opérateurs »

costume #

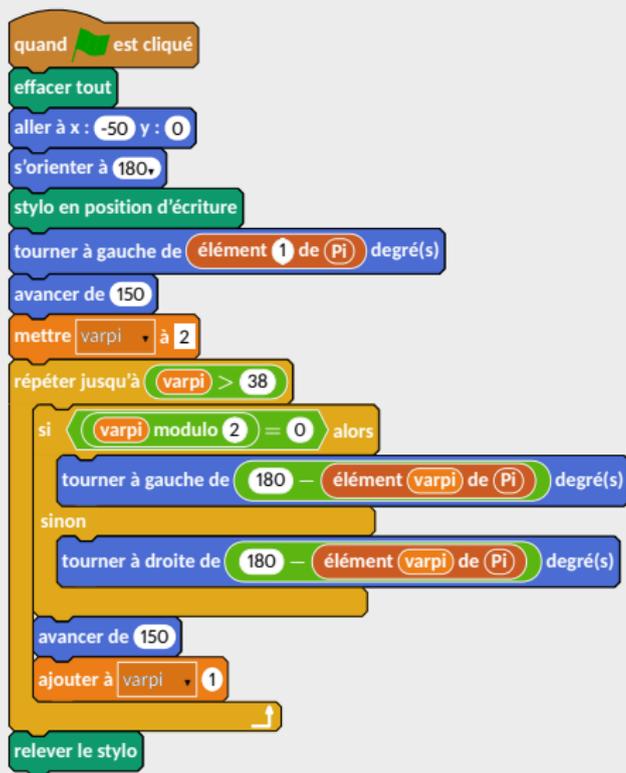
nom de l'arrière-plan

taille

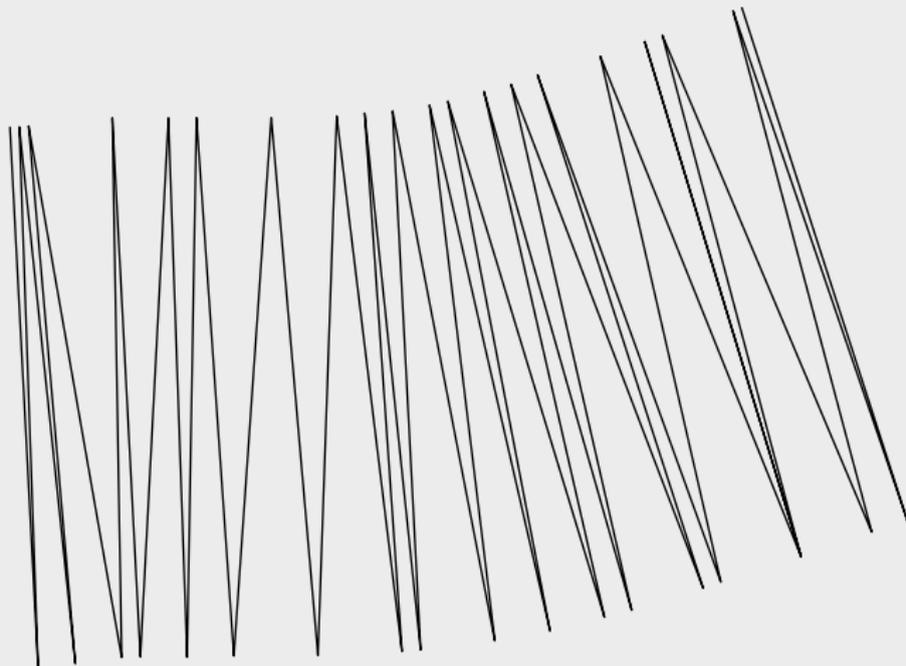
arrière-plan #

s'obtiennent avec la commande OvalApp().

À vous de jouer !



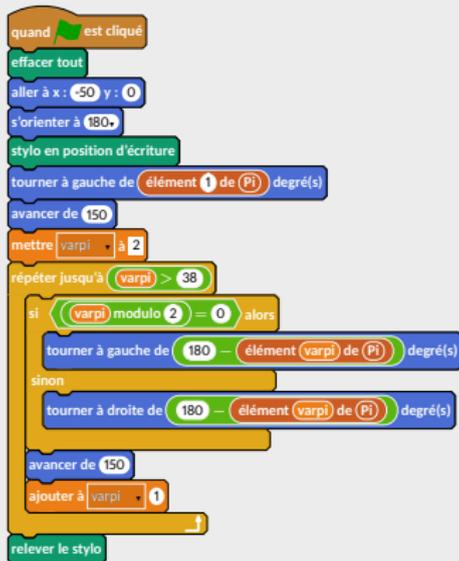
À vous de jouer !



```

draw Drapeau;
draw Effacer;
draw Aller("-50", "0");
draw Orienter("180");
draw PoserStylo;
picture BB[];
BB1=OvalListMulti("élément", OvalNb("1"), "de",
  OvalList("Pi"));
draw Tournerg(BB1);
draw Avancer("150");
draw MettreVar("varpi", "2");
draw RepeterJ(OvalOp(OvalVar("varpi"), "$\lrm{>}$",
  OvalNb("38")));
BB2=OvalOp(OvalVar("varpi"), "modulo", OvalNb("2"));
BB3=TestOp(BB2, "$\lrm{=}$", OvalNb("0"));
draw Si(BB3);
BB4=OvalListMulti("élément", OvalVar("varpi"), "de",
  OvalList("Pi"));
draw Tournerg(OvalOp(OvalNb("180"), "$\lrm{-}$", BB4));
draw Sinon;
draw Tournerd(OvalOp(OvalNb("180"), "$\lrm{-}$", BB4));
draw FinBlocSi;
draw Avancer("150");
draw AjouterVar("varpi", "1");
draw FinBlocRepeter;
draw ReleverStylo;

```



Groupe « Son »



A screenshot of the Scratch sound block palette. The blocks are arranged vertically and include:

- jouer le son **pop** ▼
- jouer le son **pop** ▼ jusqu'au bout
- arrêter tous les sons
- jouer du tambour **1** ▼ pendant **0**
- faire une pause pour **0.25** temps
- jouer la note **60** ▼ pendant **0.5** t
- choisir l'instrument n° **1** ▼
- ajouter **-10** au volume
- mettre le volume au niveau **100**
- volume
- ajouter **20** au tempo
- mettre le tempo à **60** bpm
- tempo

Groupe « Son »

A screenshot of a Scratch script for a sound group. The script contains the following blocks:

- jouer le son **pop**
- jouer le son **pop** jusqu'au bout
- arrêter tous les sons
- jouer du tambour **1** pendant **0**
- faire une pause pour **0.25** temps
- jouer la note **60** pendant **0.5** temps
- choisir l'instrument n° **1**
- ajouter **-10** au volume
- mettre le volume au niveau **100**
- volume**
- ajouter **20** au tempo
- mettre le tempo à **60** bpm
- tempo**

- **draw** Jouer("miaou");
- **draw** JouerT("miaou");
- **draw** ArrêterSon;
- **draw** Tambour("2","0.25");
- **draw** Pause("0.25");
- **draw** JouerNote("50","0.25");
- **draw** ChoisirInstrument("17");
- **draw** AjouterVol("-10");
- **draw** MettreVol("15");
- **draw** AjouterTempo("20");
- **draw** MettreTempo("15");

Les « opérateurs »

volume

tempo

s'obtiennent par la commande OvalSon().



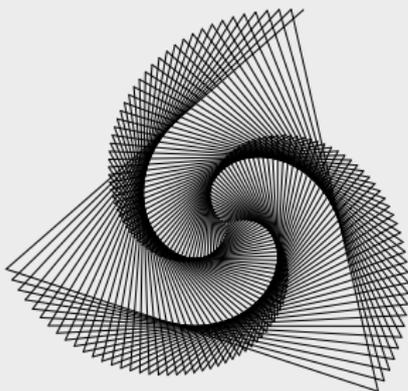
dire regroupe Le volume actuel est Volume pendant 2 seconde(s)

```
draw DireT (OvalOp ("regroupe" , RecText ("Le volume actuel est  
" ) , OvalSon ("Volume" ) ) , "2" ) ;
```

À vous de jouer !



À vous de jouer !



```

draw Drapeau;
draw ReleverStylo;
draw Aller ("0", "0");
draw Orienter ("90");
draw PoserStylo;
draw MettreVar ("i", "1");
draw RepeterJ (TestOp (OvalVar ("i"), "_$\\lmbd{=}$_", OvalNb ("200")));
draw Avancer (OvalVar ("i"));
draw AjouterVar ("i", "1");
draw Tourner ("121");
draw FinBlocRepete;
  
```

Groupe « Stylo »

effacer tout

estampiller

stylo en position d'écriture

relever le stylo

mettre la couleur du stylo à 

ajouter 10 à la couleur du stylo

mettre la couleur du stylo à 0

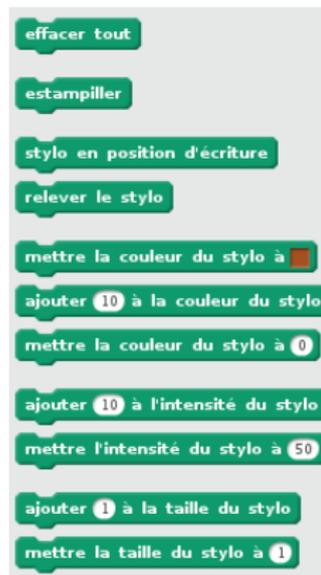
ajouter 10 à l'intensité du stylo

mettre l'intensité du stylo à 50

ajouter 1 à la taille du stylo

mettre la taille du stylo à 1

Groupe « Stylo »



- **draw** Effacer; effacer tout
- **draw** Estampiller; estampiller
- **draw** PoserStylo; stylo en position d'écriture
- **draw** ReleverStylo; relever le stylo
- **draw** MettreCouleur("Magenta",1,0,1); mettre la couleur du stylo à [Magenta]
- **draw** MettreCS("25"); mettre la couleur du stylo à [25]
- **draw** MettreIS("15"); mettre l'intensité du stylo à [15]
- **draw** AjouterTS("12"); ajouter [12] à la taille du stylo
- **draw** AjouterCS("15"); ajouter [15] à la couleur du stylo

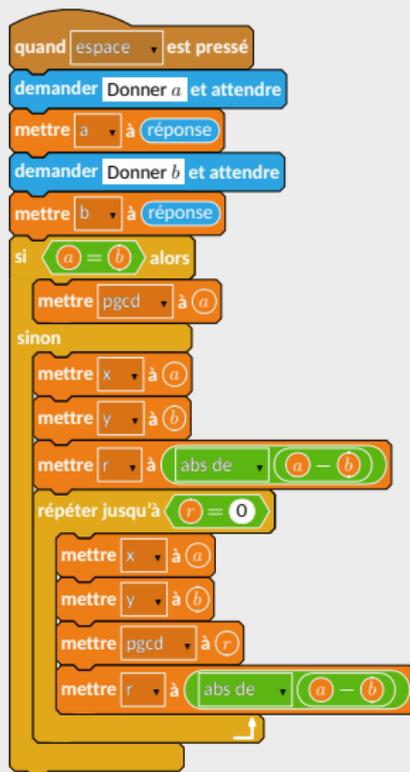
- **draw** AjouterIS(OvalOp(OvalNb("15"),"\$\bm{-}\$",OvalNb("10")));

ajouter [15 - 10] à l'intensité du stylo

- **draw** MettreTS(OvalOp(OvalNb("15"),"\$\bm{\times}\$",OvalNb("10")));

mettre la taille du stylo à [15 × 10]

À vous de jouer !



À vous de jouer !

```
draw QPresse("espace");
draw Demander("Donner_$$");
draw MettreVar("a",OvalCap("réponse"));
draw Demander("Donner_$$");
draw MettreVar("b",OvalCap("réponse"));
draw Si(TestOp(OvalVar("$$"),"$\bm{=}$$",OvalVar("$$")));
draw MettreVar("pgcd",OvalVar("$$"));
draw Sinon;
draw MettreVar("x",OvalVar("$$"));
draw MettreVar("y",OvalVar("$$"));
picture BB[];
BB1=OvalOp(OvalVar("$$"),"$\bm{-}$$",OvalVar("$$"));
draw MettreVar("r",OvalOp(RecMenuOp("abs de"),BB1));
draw RepeterJ(TestOp(OvalVar("$$"),"$\bm{=}$$",OvalNb("0")));
draw MettreVar("x",OvalVar("$$"));
draw MettreVar("y",OvalVar("$$"));
draw MettreVar("pgcd",OvalVar("$$"));
draw MettreVar("r",OvalOp(RecMenuOp("abs de"),BB1));
draw FinBlocRepeter;
draw FinBlocSi;
```

Groupe « Données »- Variables



Groupe « Données »- Variables



- **draw** MettreVar("pi", "0");
- **draw** AjouterVar("pi", "1");
- **draw** MontrerVar("pi");
- **draw** CacherVar("pi");



Groupe « Données »- Listes



Groupe « Données »- Listes



- **draw** AjouterList(RecText("\\LaTeX"),"Listepi");

ajouter \LaTeX à Listepi

- **draw** SupprimerList(RecText("\\LaTeX"),"Listepi");

supprimer l'élément \LaTeX de la liste Listepi

- **draw** InsérerList(RecText("\\MP"),"1"," Listepi ");

insérer METAPOST en position 1 de la liste Listepi

- **draw** RemplacerList(3,"Listepi",OvalOp(OvalNb("4")," $\$\\bm{+}\\$"$),OvalNb("5"));

remplacer l'élément 3 de la liste Listepi par 4 + 5

- **draw** MontrerList("Listepi");

montrer la liste Listepi

- **draw** CacherList("Listepi");

cacher la liste Listepi

Les « opérateurs »



s'obtiennent par les commandes `\OvalVar()` et `\OvalListMulti()`.



```
draw OvalListMulti (RecMenuList ("Pythagore") , "
contient_" , RecText (" (3;4;5)_?" ) ) ;
```

À vous de jouer !









```

draw Drapeau;
draw Effacer;
draw MettreTS ("1");
draw Repeter ("6");
draw PoserStylo;
draw Repeter ("4");
draw Avancer ("50");
draw Tournerd ("90");
draw FinBlocRepeter;
draw ReleverStylo;
draw AjouterTS ("1");
draw Avancer ("60");
draw FinBlocRepeter;

```

Groupe « Évènements »



Groupe « Évènements »



- **draw** Drapeau;
- **draw** QPresse("espace");
- **draw** QLutinPresse;
- **draw** QBasculeAR("arriere-plan1");

quand [drapeau] est cliqué

quand [espace] est pressé

quand ce lutin est cliqué

quand l'arrière-plan bascule sur [arrière-plan1]

- **draw** QVolumeSup("Volume_sonore",OvalNb("10"));

quand [Volume sonore] > 10

- **draw** QRecevoirMessage("message1");
- **draw** EnvoyerMessage("message1");
- **draw** EnvoyerMessageA("message1");

quand je reçois [message1]

envoyer à tous [message1]

envoyer à tous [message1] et attendre

Groupe « Évènements »



- **draw** Drapeau;

quand [drapeau] est cliqué

- **draw** QPresse("espace");

quand [espace] est pressé

- **draw** QLutinPresse;

quand ce lutin est cliqué

- **draw** QBasculeAR("arriere-plan1");

quand l'arrière-plan bascule sur [arrière-plan1]

- **draw** QVolumeSup("Volume_sonore",OvalNb("10"));

quand [Volume sonore] > 10

- **draw** QRecevoirMessage("message1");

quand je reçois [message1]

- **draw** EnvoyerMessage("message1");

envoyer à tous [message1]

- **draw** EnvoyerMessageA("message1");

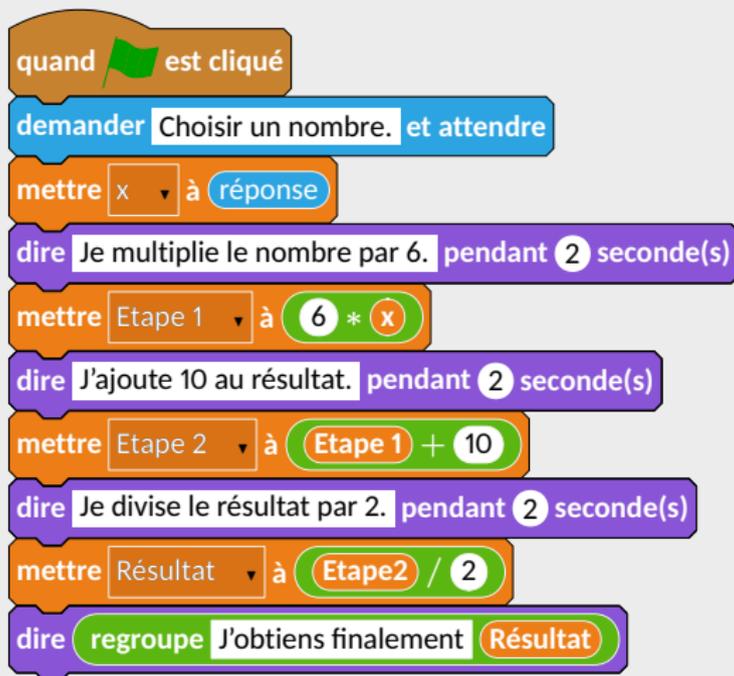
envoyer à tous [message1] et attendre

Quand « la scène » est sélectionnée, on dispose de :

- **draw** QScenePressee;

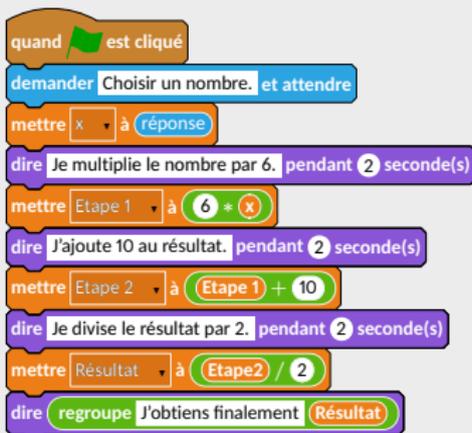
quand la Scène est cliquée

À vous de jouer !



A Scratch script for a number game. It starts with a 'when clicked' event block. The script then asks the user to choose a number and waits for the response. The response is stored in a variable 'x'. A speech bubble says 'Je multiplie le nombre par 6.' for 2 seconds. The variable 'Etape 1' is set to $6 * x$. Another speech bubble says 'J'ajoute 10 au résultat.' for 2 seconds. The variable 'Etape 2' is set to $\text{Etape 1} + 10$. A third speech bubble says 'Je divise le résultat par 2.' for 2 seconds. The variable 'Résultat' is set to $\text{Etape 2} / 2$. Finally, a 'regroupe' block displays the final result as 'Résultat'.

```
quand est cliqué  
demander Choisir un nombre. et attendre  
mettre x à réponse  
dire Je multiplie le nombre par 6. pendant 2 seconde(s)  
mettre Etape 1 à  $6 * x$   
dire J'ajoute 10 au résultat. pendant 2 seconde(s)  
mettre Etape 2 à  $\text{Etape 1} + 10$   
dire Je divise le résultat par 2. pendant 2 seconde(s)  
mettre Résultat à  $\text{Etape 2} / 2$   
dire regroupe J'obtiens finalement Résultat
```



```

draw Drapeau;
draw Demander(" Choisir_un_nombre. ");
draw MettreVar("x",OvalCap(" réponse"));
draw DireT("Je_multiplie_le_nombre_par_6.", "2");
draw MettreVar("Etape_1",OvalOp(OvalNb("6"), "_$\\bm{+}$_",OvalVar("x")));
draw DireT("J' ajoute_10_au_résultat.", "2");
draw MettreVar("Etape_2",OvalOp(OvalVar("Etape_1"), "_$\\bm{+}$_",OvalNb("10")));
draw DireT("Je_divise_le_résultat par 2.", "2");
draw MettreVar("Résultat",OvalOp(OvalVar("Etape2"), "_$\\bm{/}$_",OvalNb("2")));
draw Dire(OvalOp("regroupe",RecText("J' obtiens_finalement"),OvalVar("Résultat")));
  
```

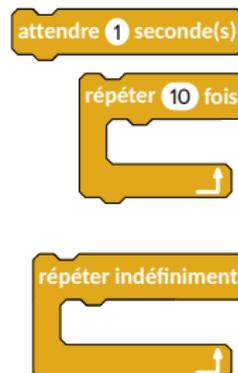
Groupe « Contrôle »



Groupe « Contrôle »



- **draw** Attendre("1");
- **draw** Repeter(10);
- **draw** LigneVide;
- **draw** FinBlocRepeter;
- **draw** RepeterI;
- **draw** LigneVide;
- **draw** FinBlocRepeterI;



Groupe « Contrôle »



- **draw** Attendre("1");
- **draw** Répéter(10);
- **draw** LigneVide;
- **draw** FinBlocRépéter;
- **draw** RépéterI;
- **draw** LigneVide;
- **draw** FinBlocRépéterI;



- **draw** Si (TestOp (OvalVar ("x"),
 " _ \$\ \bm{=} \$ _ " ,OvalNb ("20"
 "))) ;
draw LigneVide ;
draw FinBlocSi ;

Groupe « Contrôle »



- **draw** Attendre("1");



- **draw** Répéter(10);



- **draw** LigneVide;
draw FinBlocRépéter;



- **draw** Répéter1;

- **draw** LigneVide;
draw FinBlocRépéter1;

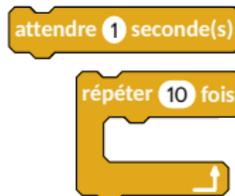
- **draw** Si (TestOp (OvalVar ("x"),
"_{\bm{=}}\$_", OvalNb ("20"
")));
draw LigneVide;
draw FinBlocSi;



Groupe « Contrôle »



- **draw** Attendre("1");
- **draw** Répéter(10);
- **draw** LigneVide;
- **draw** FinBlocRépéter;



- **draw** Répéter1;
- **draw** LigneVide;
- **draw** FinBlocRépéter1;



- **draw** Si (TestOp (OvalVar ("x"),
" ${}_{}$ ", OvalNb ("20"
")));
- **draw** LigneVide;
- **draw** Sinon;
- **draw** LigneVide;
- **draw** FinBlocSi;



Groupe « Contrôle »



Groupe « Contrôle »

- **draw** AttendreJ(TestOp(OvalVar("x"), "_\$\\bm{=}\$_", OvalNb("20")))

attendre jusqu'à 

- **draw** RepeterJ (TestOp (OvalVar (" x ") , " _ \$ \\bm { = } \$ _ " , OvalNb (" 20 "))) ;
- **draw** LigneVide ;
- **draw** FinBlocRepete ;



- **draw** Stop("ce_script");
- **draw** CommencerClone;
- **draw** CréerClone("Lutin1");
- **draw** SupprimerClone;

répéter jusqu'à 

stop ce script

quand je commence comme un clone

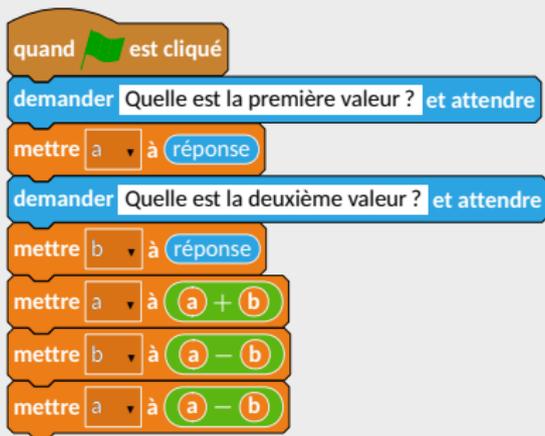
créer un clone de Lutin1

supprimer ce clone

À vous de jouer !

A Scratch script for a simple calculator. It starts with a 'when clicked' event block. The script then asks for the first value, stores it in variable 'a', asks for the second value, stores it in variable 'b', and finally calculates the sum, difference, and product of 'a' and 'b'.

```
when green flag is clicked
  ask "Quelle est la première valeur ?" and wait
  set a to response
  ask "Quelle est la deuxième valeur ?" and wait
  set b to response
  set a to a + b
  set b to a - b
  set a to a * b
```



```

draw Drapeau ;
draw Demander ("Quelle est la première valeur ?") ;
draw MettreVar ("a" ,OvalCap ("réponse")) ;
draw Demander ("Quelle est la deuxième valeur ?") ;
draw MettreVar ("b" ,OvalCap ("réponse")) ;
picture BB[];
BB1=OvalOp (OvalVar ("a" ) , " $\\bm{+}$ " ,OvalVar ("b" )) ;
BB2=OvalOp (OvalVar ("a" ) , " $\\bm{-}$ " ,OvalVar ("b" )) ;
draw MettreVar ("a" ,BB1) ;
draw MettreVar ("b" ,BB2) ;
draw MettreVar ("a" ,BB2) ;
  
```

Groupe « Capteurs »



A screenshot of the Scratch 'Capteurs' (Sensors) menu. The menu items are as follows:

- pointeur de souris ▼ touche?
- couleur [] touchée?
- couleur [] touche [] ?
- distance de [] pointeur de souris ▼
- demander [What's your name?] et att
- réponse
- touche [espace] pressée?
- souris pressée?
- souris x
- souris y
- volume sonore
- video mouvement - sur ce lutin
- activer la video [Active ▼]
- mettre la transparence vidéo à []
- chronometre
- réinitialiser le chronometre
- abscisse x ▼ de [Lutin1 ▼]
- actuel [minute ▼]
- jours depuis 2000
- nom d'utilisateur

Groupe « Capteurs »



- **draw** Demander("Quel_est_votre_nom_?");

demande Quel est votre nom ? et attendre

- **draw** ActiverVideo("activ'e");

activer la vidéo activé ▾

- **draw** TransparenceVideo("15");

mettre la transparence vidéo à 15 %

- **draw** ReinitChrono;

réinitialiser le chronomètre

Groupe « Capteurs »



- **draw** Demander("Quel_est_votre_nom_?");

demander Quel est votre nom ? et attendre

- **draw** ActiverVideo("activ\`e");

activer la vidéo activé ▾

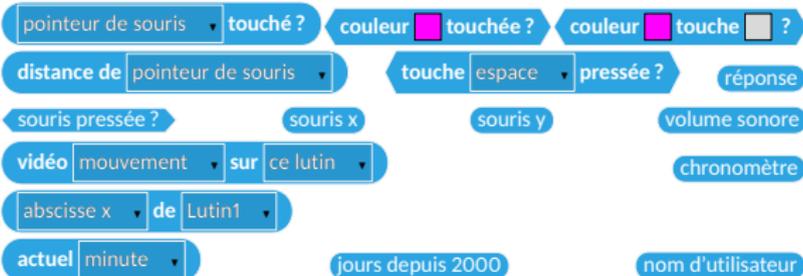
- **draw** TransparenceVideo("15");

mettre la transparence vidéo à 15 %

- **draw** ReinitChrono;

réinitialiser le chronomètre

Les « opérateurs »



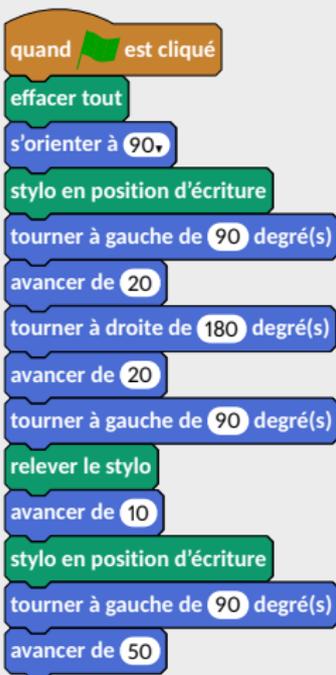
s'obtiennent par les commandes OvalCap et TestCap.

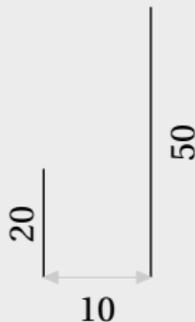
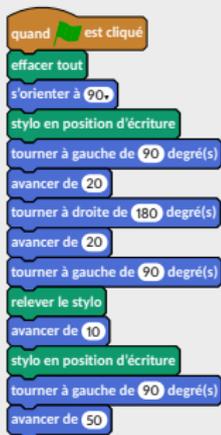
couleur  touche  ?



```
draw TestCapMulti ("couleur" ,RecCouleur (1 ,0 ,1) , "touche" ,  
  RecCouleur (0.85 ,0.85 ,0.85) , "?" ) ;
```

À vous de jouer !





```

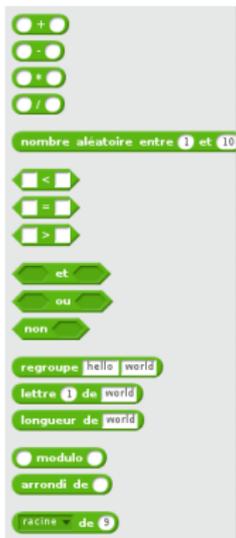
draw Drapeau;
draw Effacer;
draw Orienter("90");
draw PoserStylo;
draw Tourner("90");
draw Avancer("20");
draw Tourner("180");
draw Avancer("20");
draw Tourner("90");
draw ReleverStylo;
draw Avancer("10");
draw PoserStylo;
draw Tourner("90");
draw Avancer("50");

```

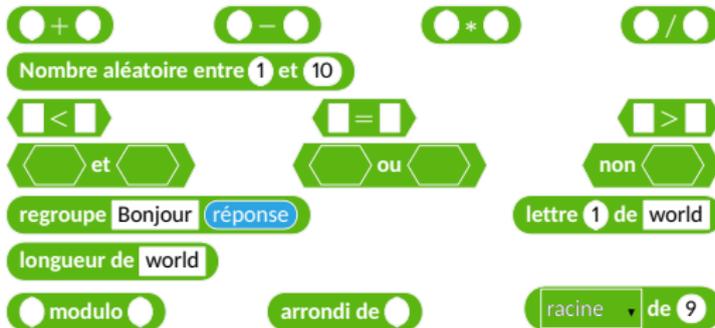
Groupe « Opérateurs »



Groupe « Opérateurs »

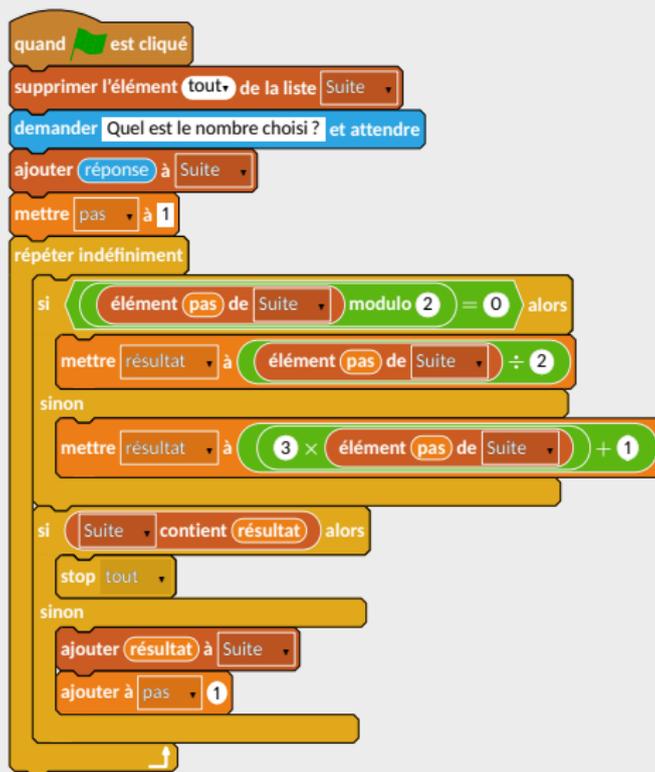


Les éléments



s'obtiennent par les commandes OvalOp() et TestOp().

À vous de jouer !



```
draw Drapeau;
draw SupprimerListe("tout", "Suite");
draw Demander("Quel_est_le_nombre_choisi?");
draw AjouterListe(OvalCap("réponse"), "Suite");
draw MettreVar("pas", "1");
draw RepeterI;
picture BB[];
BB1=OvalListMulti("élément", OvalVar("pas"), "_de_", RecMenuList("Suite"));
BB2=OvalOp(BB1, "_modulo_", OvalNb("2"));
BB3=TestOp(BB2, "_$\\bm{=} $" , OvalNb("0"));
BB4=OvalOp(BB1, "_$\\bm{\\ div} $" , OvalNb("2"));
BB5=OvalOp(OvalNb("3"), "_$\\bm{\\ times} $" , BB1);
BB6=OvalOp(BB5, "_$\\bm{+} $" , OvalNb("1"));
draw Si(BB3);
draw MettreVar("résultat", BB4);
draw Sinon;
draw MettreVar("résultat", BB6);
draw FinBlocSi;
BB7=OvalListMulti(RecMenuList("Suite"), "_contient_" , OvalVar("résultat"));
draw Si(BB7);
draw Stop("tout");
draw Sinon;
draw AjouterListe(OvalVar("résultat"), "Suite");
draw AjouterVar("pas", "1");
draw FinBlocSi;
draw FinBlocRepeter;
```

Groupe « Ajouter blocs »

Groupe « Ajouter blocs »

- **draw** NouveauBloc("Pentagone");



- **draw** NouveauBloc("Pentagone",OvalBloc("cote"));



- **draw** Bloc("Pentagone");



À vous de jouer !

```

définir la taille du stylo est aléatoire
mettre la taille du stylo à nombre aléatoire entre 3 et 12
  
```

①

```

définir la couleur du stylo est aléatoire
mettre la couleur du stylo à nombre aléatoire entre 1 et 200
  
```

②

③

④

```

effacer tout
mettre la couleur du stylo à [ ]
mettre la taille du stylo à 10
répéter 24 fois
  Point
  avancer de 20
  tourner à droite de 15 degré(s)
  
```

```

effacer tout
mettre la couleur du stylo à [ ]
la taille du stylo est aléatoire
répéter 24 fois
  Point
  avancer de 20
  tourner à droite de 15 degré(s)
  
```

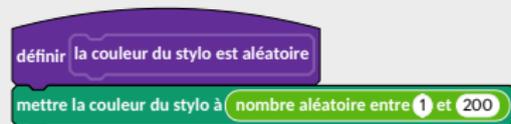
```

effacer tout
mettre la couleur du stylo à [ ]
répéter 24 fois
  la taille du stylo est aléatoire
  Point
  avancer de 20
  tourner à droite de 15 degré(s)
  
```

```

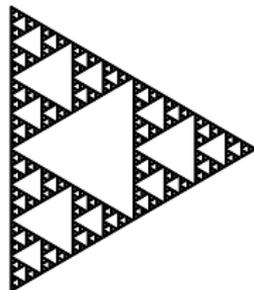
effacer tout
mettre la couleur du stylo à [ ]
répéter 24 fois
  la taille du stylo est aléatoire
  la couleur du stylo est aléatoire
  Point
  avancer de 20
  tourner à droite de 15 degré(s)
  
```

À vous de jouer !

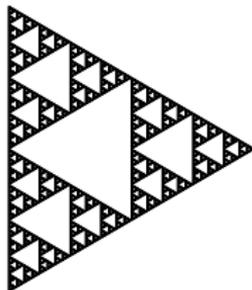


```
draw NouveauBloc ("la taille du stylo est aléatoire") ;  
draw MettreTS (OvalOp ("nombre aléatoire entre " ,OvalNb ("3" ) , " et " ,OvalNb ("12" ))) ;  
endfig ;
```

Avec plusieurs éléments ?



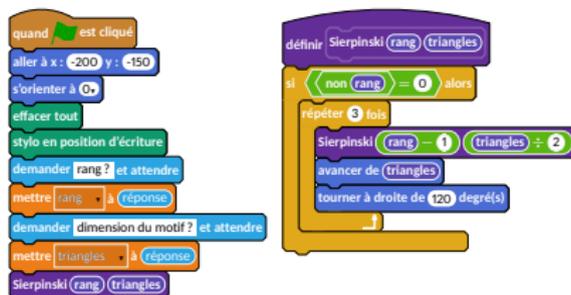
Avec plusieurs éléments ?



```

quand est cliqué
  aller à x : -200 y : -150
  s'orienter à 0°
  effacer tout
  stylo en position d'écriture
  demander rang ? et attendre
  mettre rang à réponse
  demander dimension du motif ? et attendre
  mettre triangles à réponse
  Sierpinski rang triangles

définir Sierpinski rang triangles
  si non rang = 0 alors
    répéter 3 fois
      Sierpinski rang - 1 triangles ÷ 2
      avancer de triangles
      tourner à droite de 120 degré(s)
  
```



```

draw Drapeau;
draw Aller ("-200", "-150"); draw Orienter ("0");
draw Effacer; draw PoserStyle;
draw Demander ("rang?");
draw MettreVar ("rang", OvalCap ("réponse"));
draw Demander ("dimension du motif?");
draw MettreVar ("triangles", OvalCap ("réponse"));
draw Bloc ("Sierpinski", OvalBloc ("rang"), OvalBloc ("triangles"));
_coinprec := (8.5cm, 0);
draw NouveauBloc ("Sierpinski", OvalBloc ("rang"), OvalBloc ("triangles"));
picture BB[];
BB1=TestOp ("non", OvalBloc ("rang"));
BB2=TestOp (BB1, "=", OvalNb ("0"));
draw Si (BB2);
draw Repeter ("3");
BB3=OvalOp (OvalBloc ("rang"), "-", OvalNb ("1"));
BB4=OvalOp (OvalBloc ("triangles"), "÷", OvalNb ("2"));
draw Bloc ("Sierpinski", BB3, BB4);
draw Avancer (OvalBloc ("triangles"));
draw Tourner ("120");
draw FinBlocRepeter; draw FinBlocSi;

```

Ce que **Scratch** n'a pas...

- Changer la couleur des lignes des blocs

Ce que **Scratch** n'a pas...

- Changer la couleur des lignes des blocs



Ce que **Scratch** n'a pas...

- Changer la couleur des lignes des blocs



```
beginfig (1);  
  draw RepeterI;  
  draw Dire ("Denis_est_le_meilleur_! :)");  
  draw FinBlocRepeterI;  
endfig;
```

```
CoulLignes:=0.9 white;
```

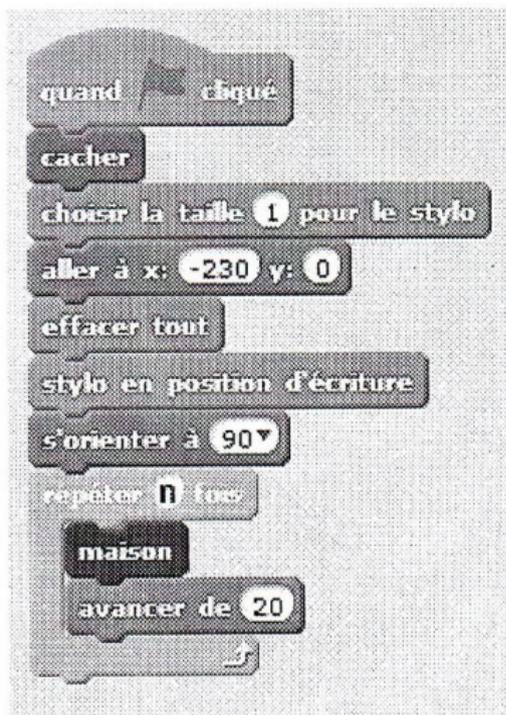
```
beginfig (2);  
  draw RepeterI;  
  draw Dire ("Denis_est_le_meilleur_! :)");  
  draw FinBlocRepeterI;  
endfig;
```

Ce que **Scratch** n'a pas...

- La possibilité d'adapter le contenu des blocs

Ce que **Scratch** n'a pas...

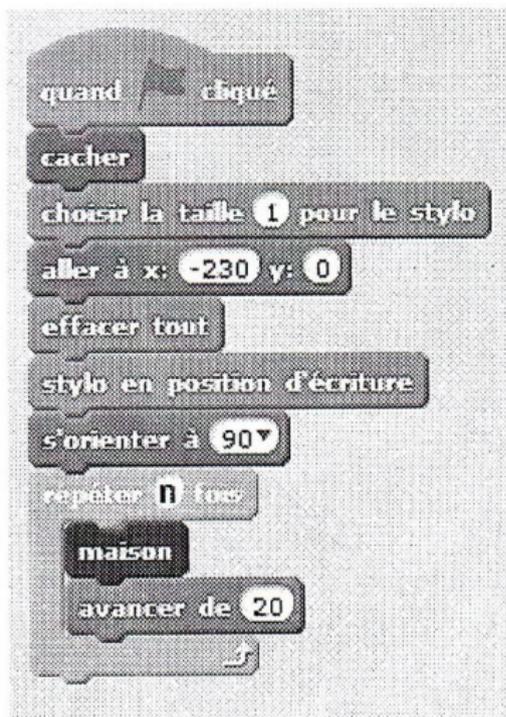
- La possibilité d'adapter le contenu des blocs



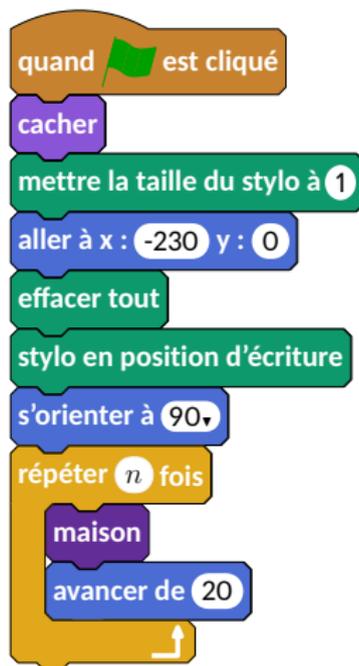
programme principal

Ce que **Scratch** n'a pas...

- La possibilité d'adapter le contenu des blocs



programme principal



Ce que **Scratch** n'a pas...

- Les lignes vides



Ce que **Scratch** n'a pas...

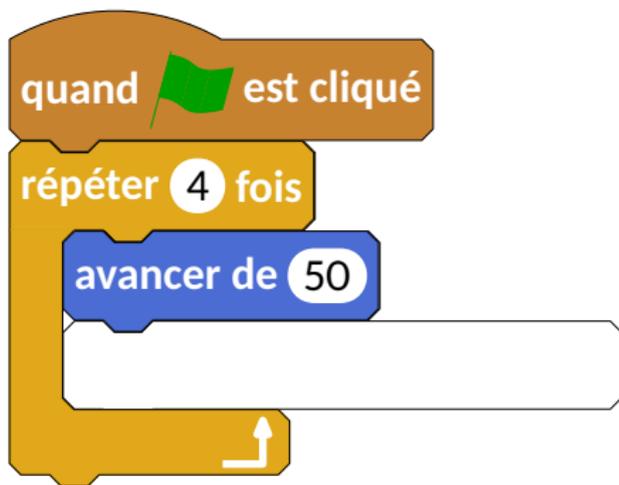
- Les lignes vides



```
beginfig (5) ;  
  draw Si (TestOp (OvalVar ("x") , "_$\\bm{=}$_" , OvalNb ("20" ) ) ) ;  
  draw LigneVide ;  
  draw Sinon ;  
  draw LigneVide ;  
  draw FinBlocSi ;  
endfig ;
```

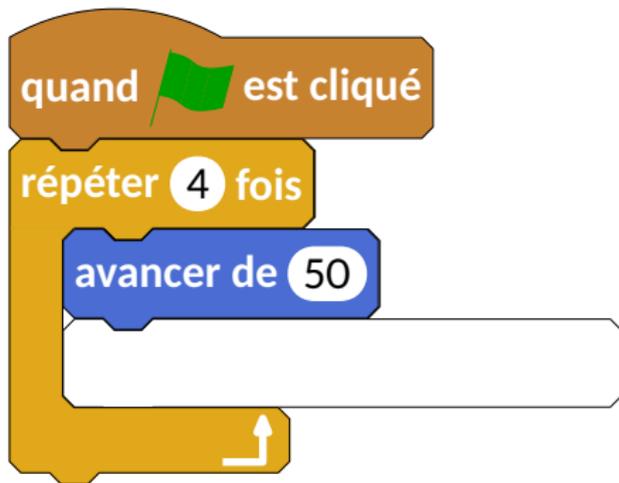
Ce que **Scratch** n'a pas...

- Les blocs vides



Ce que **Scratch** n'a pas...

- Les blocs vides



```
draw Drapeau ;  
draw Repeter ("4") ;  
draw Avancer ("50") ;  
draw CommandeVide ("5") ;  
draw FinBlocRepeter ;
```

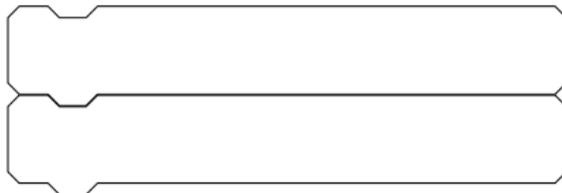
Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- Les lignes pointillés



...

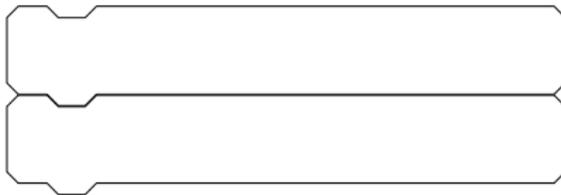


Ce que **Scratch** n'a pas...

- Les lignes pointillés



...

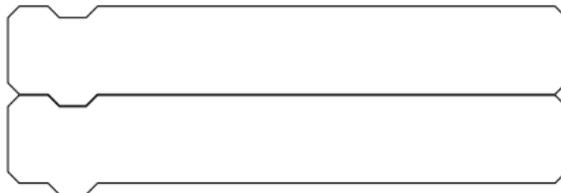


Ce que **Scratch** n'a pas...

- Les lignes pointillés



...



draw Drapeau ;
draw LignePointilles ;
draw CommandeVide("5") ;
draw CommandeVide("5") ;

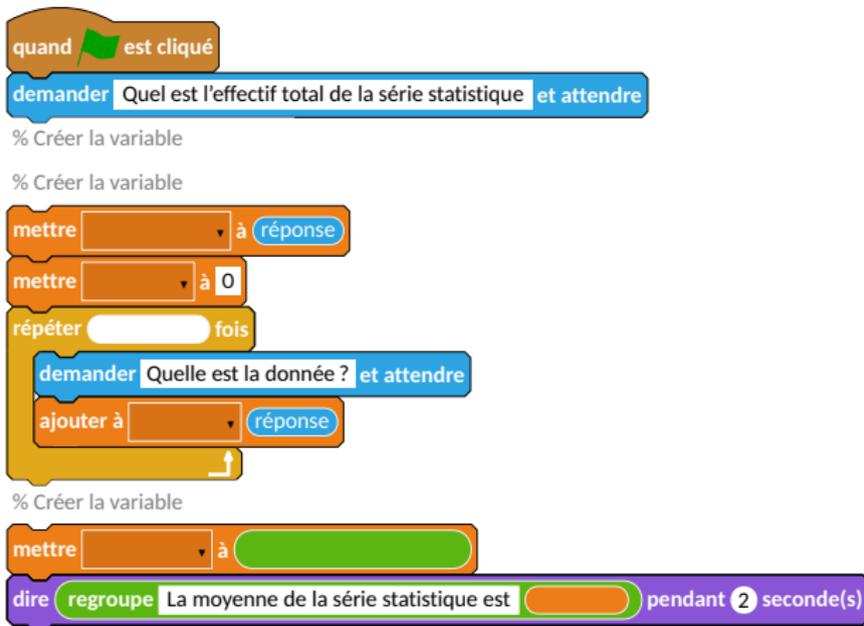
Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- Les commentaires « en continu »

Ce que **Scratch** n'a pas...

- Les commentaires « en continu »



Scratch script illustrating the use of comments:

- when green flag clicked
- ask "Quel est l'effectif total de la série statistique" and wait
- % Créer la variable
- % Créer la variable
- set [] to réponse
- set [] to 0
- repeat [] fois
 - ask "Quelle est la donnée ?" and wait
 - add [] to réponse
- % Créer la variable
- set [] to []
- say "regroupe La moyenne de la série statistique est []" for 2 seconds

Ce que **Scratch** n'a pas...

```
draw Drapeau ;
draw Demander("Quel est l'effectif total de la série statistique ?") ;
draw Commentaires("% Créer la variable") ;
draw Commentaires("% Créer la variable") ;
draw MettreVar("\phantom{Effectif total}", OvalCap("réponse")) ;
draw MettreVar("\phantom{Somme}", "0") ;
draw Repeter(OvalNb("\phantom{Effectif total}")) ;
draw Demander("Quelle est la donnée ?") ;
draw AjouterVar("\phantom{Somme}", OvalCap("réponse")) ;
draw FinBlocRepeter ;
draw Commentaires("% Créer la variable") ;
draw MettreVar("\phantom{Moyenne}", OvalOp("\phantom{\$ \mbox{Somme} \bmn{\div} \mbox{Effectif total}}$")) ;
draw DireT(OvalOp("regroupe", RecText("La moyenne de la série statistique est
"), OvalVar("\phantom{Moyenne}")), "2") ;
```

Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- Les commentaires

Ce que **Scratch** n'a pas...

- Les commentaires en fin de ligne

Ce que **Scratch** n'a pas...

- Les commentaires en fin de ligne



Ce que **Scratch** n'a pas...

- Les commentaires en fin de ligne



```

draw CommentairesLigne (" \footnotesize On lance le script par ce bloc " );
draw Avancer ("50" );
draw CommentairesLigne (" \footnotesize Le lutin va-t-il tracer un segment ? " );
draw Tournerd ("90" );
draw CommentairesLigne (" \footnotesize \begin { minipage } { 150 pt } Est-ce le bon angle pour obtenir un hexagone
régulier ? \end { minipage } " );
draw Avancer ("50" );
draw Tournerd ("90" );
draw CommentairesLigne (" \footnotesize On ne peut pas calculer  $\int_1^2 x^2 dx$  ? " );

```

Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- La numérotation des lignes

Ce que **Scratch** n'a pas...

- La numérotation des lignes

```
1 quand est cliqué %Créer les variables Adultes et Bébés
2 mettre Adultes à 2
3 mettre Bébés à 0
4 répéter 73 fois
5   ajouter à Adultes Bébés
6   si (Adultes modulo 2 = 0) alors %On teste si le nombre d'adultes est pair
7     mettre Bébés à (Adultes ÷ 2)
8   sinon
9     mettre Bébés à ((Adultes - 1) ÷ 2)
10  dire Le total de vaches au bout de 24 heures est...
11  dire (Adultes + Bébés) pendant 10 seconde(s)
```

Ce que **Scratch** n'a pas...

```
draw Drapeau;
draw CommentairesLigne("%Créer les variables \textbf{Adultes} et \textbf{Bébés}");
draw MettreVar("Adultes", "2");
draw MettreVar("Bébés", "0");
draw Repeter("73");
picture BB[];
BB1=OvalVar("Bébés");
BB2=OvalOp(OvalVar("Adultes"), "modulo", OvalNb("2"));
BB3=TestOp(BB2, "$\bm{=}$", RecText("0"));
BB4=OvalOp(OvalVar("Adultes"), "$\bm{\div}$", OvalNb("2"));
BB5=OvalOp(OvalVar("Adultes"), "$\bm{-}$", OvalNb("1"));
BB6=OvalOp(BB5, "$\bm{\div}$", OvalNb("2"));
BB7=OvalOp(OvalVar("Adultes"), "$\bm{+}$", OvalVar("Bébés"));
draw AjouterVar("Adultes", BB1);
draw Si(BB3);
draw CommentairesLigne("%On teste si le nombre d'adultes est pair");
draw MettreVar("Bébés", BB4);
draw Sinon;
draw MettreVar("Bébés", BB6);
draw FinBlocSi;
draw FinBlocRepeter;
draw Dire("Le total de vaches au bout de 24 heures est...");
draw DireT(BB7, "10");
```

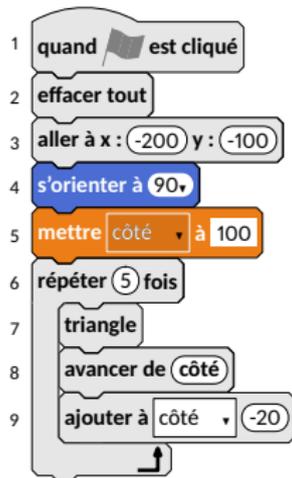
Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- L'impression

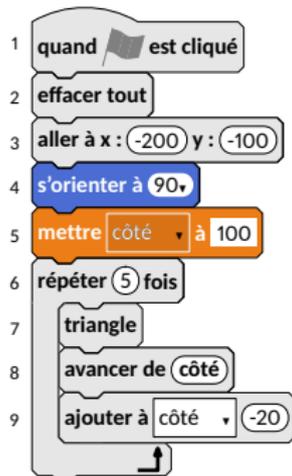
Ce que **Scratch** n'a pas...

- L'impression



Ce que **Scratch** n'a pas...

- L'impression



```

coefprint:=0.9;
print:=true;
  
```

```
NumeroteLignes:=true;
```

```

draw Drapeau;
draw Effacer;
draw Aller("-200","-100");
  
```

```
print:=false;
```

```

draw Orienter("90");
draw MettreVar("côté","100");
  
```

```
print:=true;
```

```

draw Repeter("5");
draw Bloc("triangle");
draw Avancer(OvalBloc("côté"));
draw AjouterVar("côté","-20");
draw FinBlocRepeter;
  
```

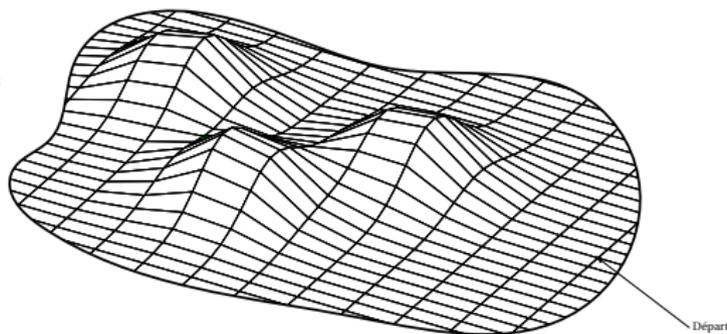
Ce que **Scratch** n'a pas...

Ce que **Scratch** n'a pas...

- Les blocs personnalisables

Ce que **Scratch** n'a pas...

- Les blocs personnalisables



Avancer de ③ vers l'ouest

Avancer de ⑧ vers le sud

Avancer de ⑤ vers l'ouest

Avancer de ③ vers le nord

Avancer de ③ vers l'est

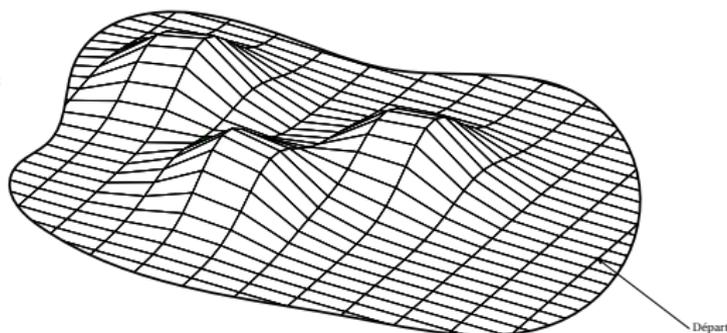
Avancer de ⑦ vers le nord

Avancer de ④ vers l'ouest

Avancer de ② vers le sud

Ce que **Scratch** n'a pas...

- Les blocs personnalisables



Avancer de 3 vers l'ouest

Avancer de 8 vers le sud

Avancer de 5 vers l'ouest

Avancer de 3 vers le nord

Avancer de 3 vers l'est

Avancer de 7 vers le nord

Avancer de 4 vers l'ouest

Avancer de 2 vers le sud

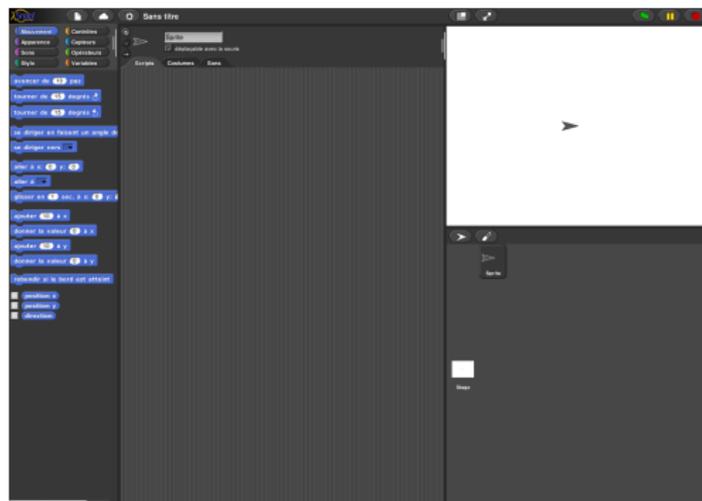
```
draw BlocUser((44/256,120/256,195/256)) ("Avancer_ de
  _",OvalNb("3"),"_vers_l'ouest");
```

Perspectives ?

- Poursuivre le développement :)

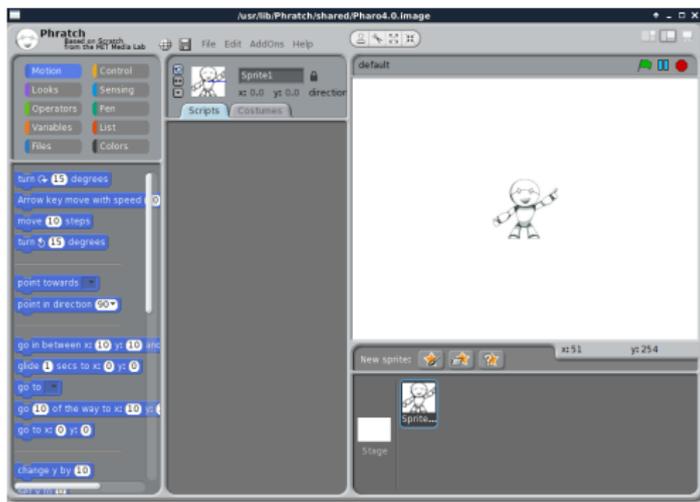
Perspectives ?

- Poursuivre le développement :)
- Snap? <http://snap.berkeley.edu/>



Perspectives ?

- Poursuivre le développement :)
- Snap ? <http://snap.berkeley.edu/>
- Phratch ? <http://www.phratch.com/>



Perspectives ?

- Poursuivre le développement :)
- Snap ? <http://snap.berkeley.edu/>
- Phratch ? <http://www.phratch.com/>
- Scratch 3 ? <https://llk.github.io/scratch-gui/custom-procedures/>

