

# Logique

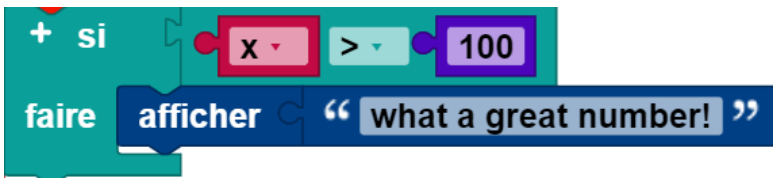
## Instructions conditionnelles

Les instructions conditionnelles sont essentielles pour la programmation. Elles permettent de formuler des distinctions de cas telles que :

- S'il y a un moyen de tourner à gauche, rotation à gauche.
- Si le score = 100, imprimer « Bien joué ! ».

### Blocs **si**

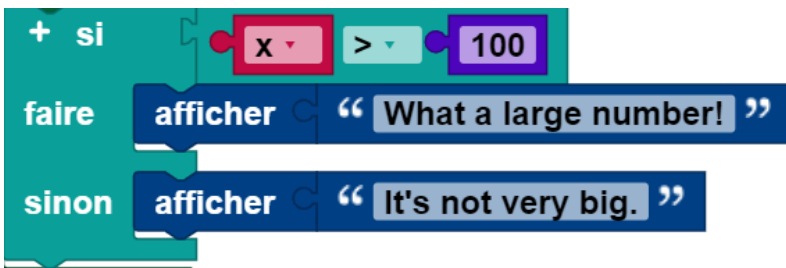
La condition la plus simple est un bloc **si** :



Lorsqu'il est exécuté, la valeur de la variable **x** est comparée à 100. Si elle est supérieure, « Quel grand nombre ! » est édité. Sinon, il ne se passe rien.

### Blocs **si-sinon**

Il est également possible d'indiquer que quelque chose doit se produire si la condition n'est pas vraie, comme dans cet exemple :

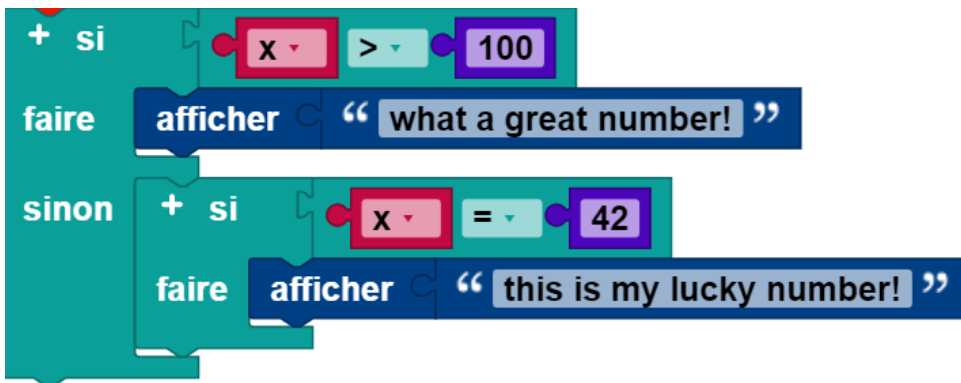


Comme pour le bloc précédent, « Quel grand nombre ! » est émis lorsque  $x > 100$ . Dans le cas contraire, « Ce n'est pas très grand » est édité.

Un bloc **si** peut avoir une section **sinon**, mais pas plus d'une.

### Blocs **si-sinon-si**

Il est également possible de tester plusieurs conditions avec un seul bloc **si** en ajoutant des clauses **sinon-si** :

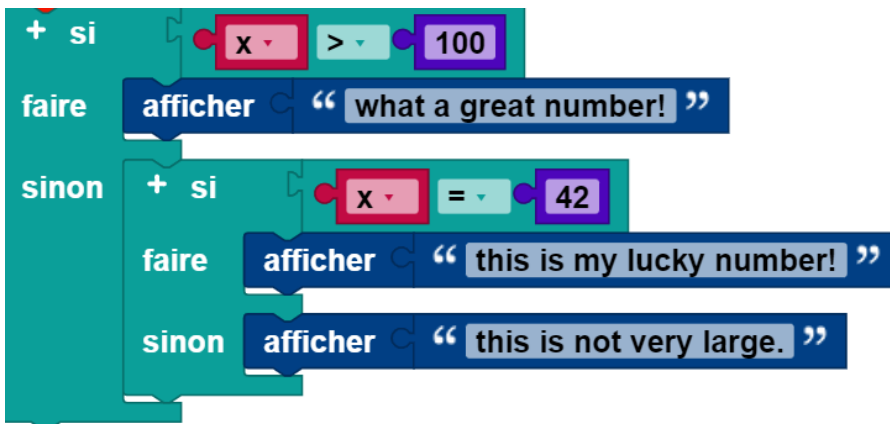


Le bloc vérifie d'abord si  $x > 100$  et émet « Quel grand nombre ! » si c'est le cas. Si ce n'est pas le cas, il continue à vérifier si  $x = 42$ . Si oui, il émet « C'est mon chiffre porte-bonheur ! ». Sinon, il ne se passe rien.

Un bloc **si** peut avoir un nombre quelconque de sections **sinon-si**. Les conditions sont évaluées de haut en bas jusqu'à ce qu'une condition soit remplie ou jusqu'à ce qu'il ne reste plus aucune condition.

## Blocs **si-sinon-si-sinon**

Les blocs **si** peuvent comporter aussi bien des sections **sinon-si** que des sections **sinon** :



La section **sinon** garantit qu'une action sera exécutée même si aucune des conditions précédentes n'est vraie.

Une autre section **sinon** peut se produire après n'importe quel nombre de sections **sinon-si**, y compris zéro, pour obtenir un bloc **si-sinon** normal.

## Modification de bloc

La barre d'outils affiche uniquement le bloc **si** simple et le bloc **si-sinon** :



Pour ajouter des clauses **si-sinon** et **sinon**, cliquez sur l'icône (+). L'icône (-) permet de supprimer à nouveau les clauses **sinon-si** :



Remarquez que les formes des blocs permettent d'ajouter un nombre quelconque de sous-blocs **sinon-si**, mais seulement jusqu'à un bloc **si**.

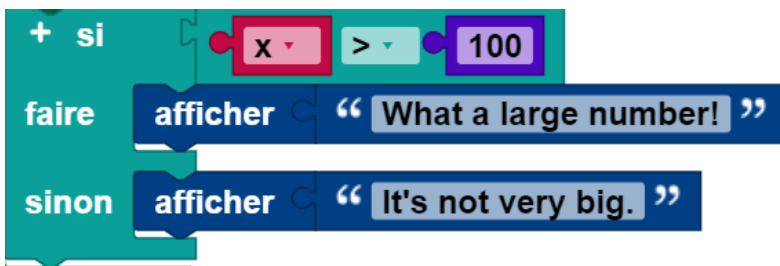
## Logique booléenne

La logique booléenne est un système mathématique simple qui a deux valeurs :

- vrai
- faux

Les blocs logiques dans ROBO Pro Coding sont généralement destinés à contrôler les conditions et les boucles.

Voici un exemple :



Si la valeur de la variable x est supérieure à 100, la condition est vraie et le texte « Quel grand nombre ! » est édité. Si la valeur de x n'est pas supérieure à 100, la condition est fausse et « Ce n'est pas très grand » est édité. Les valeurs booléennes peuvent également être stockées dans des variables et transmises à des fonctions, de

même que les nombres, le texte et les valeurs de liste.

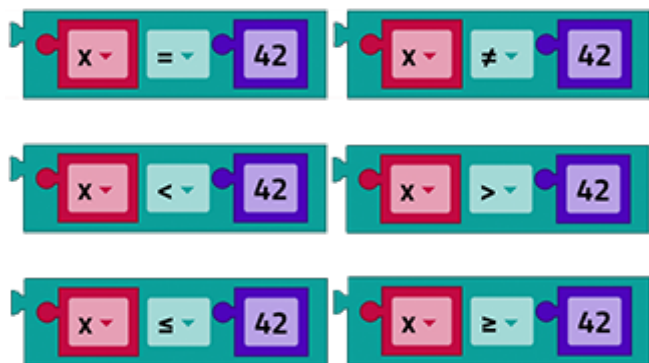
Si un bloc attend une valeur booléenne comme entrée, une entrée manquante est interprétée comme **incorrecte**. Les valeurs non booléennes ne peuvent pas être insérées directement là où des valeurs booléennes sont attendues, bien qu'il soit possible (mais non conseillé) d'enregistrer une valeur non booléenne dans une variable et de l'insérer ensuite dans l'entrée de condition. Cette méthode n'est pas recommandée et son comportement peut changer dans les versions futures de ROBO Pro Coding.

## Valeurs

Un seul bloc avec une liste déroulante indiquant soit **vrai**, soit **faux** peut être utilisé pour extraire une valeur booléenne :

## Opérateurs de comparaison

Il y a six opérateurs de comparaison. Deux entrées (normalement deux nombres) sont transmises à chacune et l'opérateur de comparaison renvoie **vrai** ou **faux**, selon la manière dont les entrées sont comparées.



Les six opérateurs sont : égal, non égal, inférieur, supérieur, inférieur ou égal, supérieur ou égal.

## Opérateurs logiques

Le bloc **et** transmet alors et seulement alors le signal **vrai** si ses deux valeurs d'entrée sont vraies.



Le bloc **ou** transmet le signal **vrai** si au moins une de ses deux valeurs d'entrée est vraie.



## pas

Le bloc **pas** transforme une saisie booléenne en sa contrepartie. Par exemple, le résultat de :



**faux**.

En l'absence de saisie, la valeur **vraie** est enregistrée, de sorte que le bloc suivant génère la valeur **faux** :



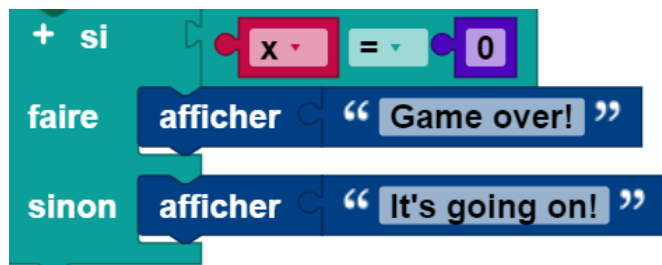
Toutefois, il n'est pas recommandé de laisser une entrée vide.

## Triple opérateur

Le triple opérateur se comporte comme un bloc **si-sinon** miniature. Il prend trois valeurs d'entrée, la première valeur d'entrée étant la condition booléenne à tester, la deuxième valeur d'entrée étant la valeur à restituer si le test est **vrai**, la troisième valeur d'entrée étant la valeur à restituer, si le test est faux. Dans l'exemple suivant, la variable **Couleur** est définie sur le rouge si la variable **x** est inférieure à 10, sinon la variable **Couleur** est définie sur le vert.



Un bloc triple peut toujours être remplacé par un bloc **si-sinon**. Les deux exemples suivants sont exactement identiques.



Revision #11

Created 17 November 2021 20:43:57 by Admin

Updated 10 November 2024 15:04:31 by phuesing