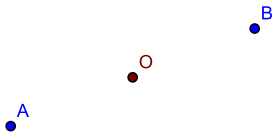
Fiche technique

35 Introduction aux langages de scripts

http://url.univ-irem.fr/ft29

QR Code

Il est possible d'enrichir les figures créées avec GeoGebra à l'aide de scripts qui permettent de déclencher certaines actions lorsque l'utilisateur modifie la figure. Les scripts autorisent, en particulier, des manipulations qu'il serait impossible d'obtenir par ailleurs. La figure, fournie en exemple ci-dessous, en est l'illustration : les points A et B sont symétriques par rapport au point O, et il est permis, ici, de déplacer l'un quelconque des trois points. Sans l'usage des scripts, le point B serait dépendant des points A et O, en supposant que l'on ait construit B comme le symétrique de A par rapport à O, et non l'inverse.



L'usage des scripts se révèle particulièrement intéressant lorsqu'il s'agit d'accéder à des fonctionnalités avancées du logiciel, ou lorsqu'on souhaite faciliter la prise en main d'un fichier en offrant un certain degré de sophistication dans l'interactivité proposée.

Un script est un petit programme écrit dans un langage particulier. GeoGebra permet d'utiliser deux langages différents : le GeoGebraScript et le JavaScript. Il n'est pas de notre ambition, dans cette fiche, de fournir un guide complet expliquant comment programmer GeoGebra ou comment exploiter le JavaScript. Nous nous contenterons, modestement, de soulever le voile sur les dessous de la programmation en langages de scripts en abordant quelques éléments fondamentaux.

Ouvrir le fichier exemple

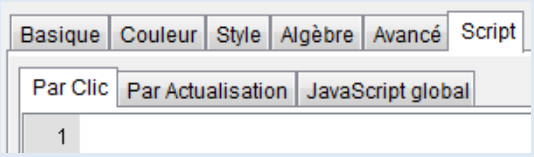
1 Programmation événementielle

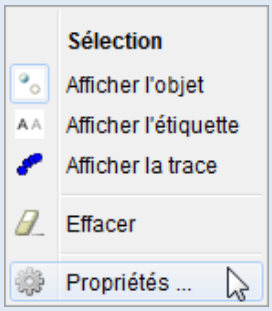
Lorsqu'on déplace un point, lorsqu'on clique sur un bouton, lorsqu'on modifie une valeur numérique, etc., on déclenche ce que l'on appelle en langage informatique, un événement. Le logiciel répond alors en fonction de l'événement survenu. Les scripts intégrés à GeoGebra permettent d'intercepter différents types d'événements et d'agir en conséquence.

Dans GeoGebra chaque objet peut se voir affecter un script.

Méthode

- Choisir un objet et effectuer un clic avec le bouton droit de la souris pour faire apparaître le menu contextuel.
- Choisir le menu Propriétés...
- Choisir l'onglet ***Script***.





L'onglet ***Script*** permet d'accéder à trois rubriques différentes qui correspondent aux différents types d'événements qu'il est possible d'intercepter :

- ***Par Clic :*** un script, écrit dans cette rubrique, sera déclenché au moment où l'utilisateur effectuera un clic avec le bouton gauche de la souris sur l'objet ;
- ***Par Actualisation :*** la moindre modification de l'objet (valeur, position, ...) déclenchera le script associé à cette rubrique ;
- ***JavaScript global :*** comme son nom l'indique, il est uniquement autorisé d'écrire des instructions en JavaScript dans cette rubrique. On peut y définir quelques fonctions globales et affecter des variables avant le chargement du fichier. C'est également dans cette rubrique qu'il est possible de définir la fonction `ggbOnInit()`, automatiquement appelée après le chargement du fichier (voir le paragraphe consacré au langage JavaScript).

Remarque :

Selon la nature de l'objet destiné à se voir affecter un script, l'événement ***Par Clic*** n'est pas nécessairement accessible. Par exemple, pour un curseur, ou encore pour un champ texte, seuls les événements ***Par Actualisation*** et ***Javascript global*** sont disponibles.

2 Le langage GeoGebraScript

Le langage GeoGebraScript possède le mérite d'être très simple à mettre en œuvre. Dans ce langage, les scripts sont constitués de commandes GeoGebra, exécutées les unes à la suite des autres.

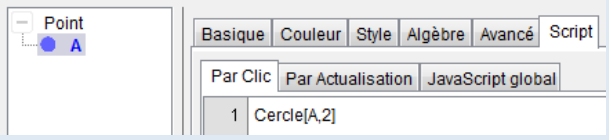
Par exemple, il est tout à fait possible de réaliser le script suivant :

Méthode

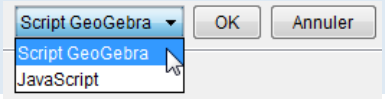
On considère un point A.

- Effectuer un clic avec le bouton droit de la souris sur le point A pour faire apparaître le menu contextuel et choisir Propriétés... .
- Dans l'onglet ***Script***, rubrique ***Par Clic***, taper le script suivant :

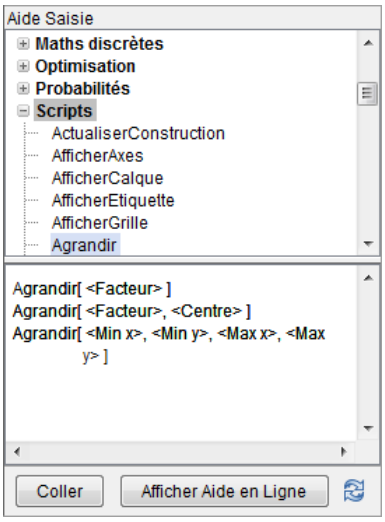
```
Cercle[A,2]
```



- Vérifier que ***Script GeoGeobra*** est sélectionné.
- Valider en cliquant sur le bouton OK.



Désormais, lorsqu'on clique sur le point A, un cercle de centre A et de rayon 2 unités est créé. Si l'utilisateur clique plusieurs fois sur le point A, plusieurs cercles (confondus) sont créés. Ce script ne présente donc aucun intérêt en soi, si ce n'est celui d'illustrer le processus de création et de fonctionnement d'un tel objet.



Les commandes utilisables dans les scripts ne se limitent pas à celles qui permettent la création d'objets (auquel cas, l'éventail des possibles serait pour le moins restreint). Fort heureusement, GeoGebra possède un ensemble de commandes (au nom souvent évocateur) dédiées à être incluses dans un script et permettant de contrôler assez finement le comportement de celui-ci. La liste complète de ces commandes est accessible depuis le panneau d'aide à la saisie (cliquer sur le bouton pour le faire apparaître), rubrique ***Scripts***.

Le lecteur, désireux de se plonger dans les arcanes de la programmation en GeoGebraScript, est invité à explorer ces différentes commandes et ne doit pas hésiter à consulter l'aide en ligne pour découvrir leur syntaxe exacte.

Bien que cette fiche n'ait pas vocation à fournir un descriptif détaillé de toutes les commandes utilisables dans un script, on peut néanmoins mentionner l'existence de la commande `SoitValeur[<objet>,<valeur>]` qui permet d'affecter la valeur `<valeur>` à l'objet `<objet>`.

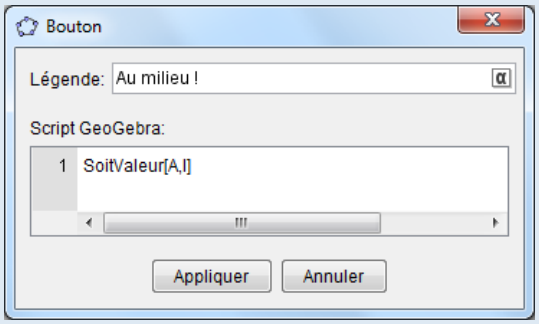
Ainsi, à l'aide de cette commande, il devient possible, par exemple, de placer un objet libre à une position donnée : imaginons un point A libre sur un segment $[MN]$ et supposons que nous souhaitions placer automatiquement le point A au milieu du segment $[MN]$, à l'appui sur un bouton.

Méthode

- Cliquer sur l'icône et construire le point I, milieu du segment $[MN]$.
- Cacher le point I (par exemple, en décochant la pastille devant le nom du point dans la fenêtre ***Algèbre***).
- Cliquer sur l'icône OK.
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la boîte de dialogue ***Bouton***.
- Compléter le champ ***Légende*** avec le texte de votre choix.
- Dans la rubrique ***Script GeoGebra***, inscrire :

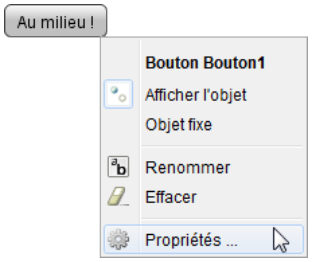
```
SoitValeur[A,I]
```

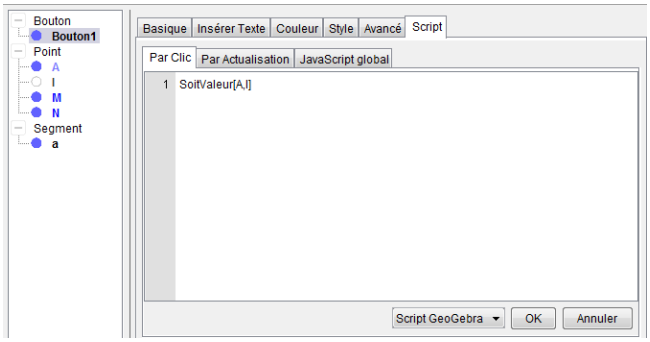
- Valider en cliquant sur le bouton Appliquer.



Remarque :

Pour modifier le script attaché à un bouton, faire apparaître le panneau des propriétés du bouton et sélectionner l'onglet ***Script***.





Une fois les modifications effectuées, ne pas oublier de les valider en cliquant sur le bouton OK

Ouvrir le fichier exemple

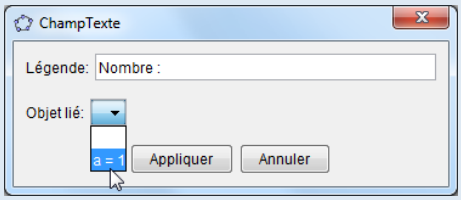
La programmation par script permet également d'interdire une valeur particulière dans un champ texte. Par exemple, supposons que nous souhaitions interdire à l'utilisateur d'entrer la valeur 0 dans un champ texte.

Méthode

- Définir une variable numérique a, en positionnant le curseur dans le champ de saisie, et en tapant, par exemple : a=1.

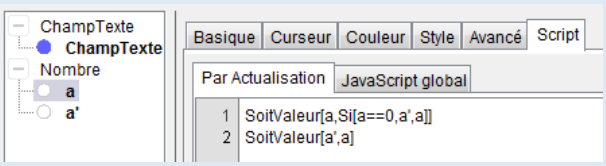
- Valider en appuyant sur la touche ↵ .
- Toujours en utilisant le champ de saisie, définir une variable numérique a', égale à la valeur initiale de a, en tapant : a'=1.

- Valider en appuyant sur la touche ↵ .
- Cliquer sur l'icône a=1.
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la boîte de dialogue ***ChampTexte***.
- Compléter le champ ***Légende*** avec le texte de votre choix.
- Dans la liste déroulante ***Objet lié***, sélectionner le nombre a.
- Valider en cliquant sur le bouton Appliquer.



- Dans la fenêtre ***Algèbre***, effectuer un clic avec le bouton droit de la souris sur la variable a pour faire apparaître le menu contextuel et choisir Propriétés... .
- Dans l'onglet ***Script***, rubrique ***Par Actualisation***, taper le script suivant :

```
SoitValeur[a,Si[a==0,a',a]]
SoitValeur[a',a]
```



- Valider en cliquant sur le bouton OK.

Ici, le champ texte est lié à la variable a, c'est-à-dire que la valeur entrée dans le champ texte est affectée à a. La variable a' sert à stocker l'ancienne valeur de a. Nous introduisons un script qui intercepte les actualisations de la variable a et qui compare alors cette nouvelle valeur à 0. Si l'utilisateur a entré 0, la variable a se voit affecter la valeur de la variable a', autrement dit, son ancienne valeur. Dans le cas contraire, on laisse inchangée la valeur de a. C'est le rôle de la commande `SoitValeur[a,Si[a==0,a',a]]`.

La commande `SoitValeur[a',a]` permet d'actualiser la valeur de a' avec la nouvelle valeur de a.

Remarque :

- Pour une description de la syntaxe de la commande `Si`, voir la fiche technique **Les valeurs booléennes**, page 593.
- Il est parfois utile de connaître la chronologie des événements lors de l'utilisation d'un script « par actualisation » :
  - lorsque l'utilisateur déplace un objet, GeoGebra met d'abord à jour les coordonnées de l'objet ;
  - ensuite, GeoGebra exécute le script associé à cet objet ;
  - enfin, GeoGebra met à jour tous les objets qui dépendent de l'objet initial.

Ouvrir le fichier exemple

3 Le langage JavaScript

La programmation en GeoGebraScript convient parfaitement pour la plupart des projets à complexité limitée. Néanmoins, ce langage, à base de commandes GeoGebra, ne permet pas de gérer des structures conditionnelles sophistiquées ou encore, des structures de données faisant intervenir, par exemple, des tableaux. Ainsi, si on souhaite une interaction poussée entre la figure et l'utilisateur, il faut recourir à la puissance du langage JavaScript.

Nous n'aborderons, dans ce paragraphe, que les spécificités de la programmation JavaScript dans GeoGebra. Des tutoriels bien plus généraux peuvent être trouvés sur le site OpenClassrooms (`http://fr.openclassrooms.com/informatique/cours/tout-sur-le-javascript`) ou encore sur ce site : `http://www.xul.fr/ecmascript`.

JavaScript est un langage de programmation orientée objet. Dans GeoGebra, les fonctions permettant d'accéder aux propriétés des éléments de la figure (ou des les modifier) sont accessibles à partir de l'objet `ggbApplet` (de telles fonctions sont appelées « méthodes »). La liste des méthodes utilisables avec l'objet `ggbApplet` est disponible sur cette page : `http://wiki.geogebra.org/fr/Référence:JavaScript`.

Prenons, par exemple, la méthode `getXcoord(<objet>)` qui renvoie l'abscisse de l'objet `<objet>` qui lui est passé en paramètre. Pour utiliser cette méthode au sein d'un script, il conviendra d'écrire `ggbApplet.getXcoord('A')`, ce qui permettra d'obtenir l'abscisse du point A.

Lorsque de nombreuses méthodes de l'objet `ggbApplet` sont utilisées, il peut rapidement devenir pénible d'être contraint d'écrire `ggbApplet` devant chacune d'elles. Pour alléger la syntaxe, il est possible d'employer l'instruction `with` qui permet de « factoriser » l'objet `ggbApplet`. Ainsi, le script

```
var a=ggbApplet.getXcoord('A');
var b=ggbApplet.getYcoord('A');
var c=ggbApplet.getValue('nombre');
```

peut s'écrire

```
with (ggbApplet) {
    var a=getXcoord('A');
    var b=getYcoord('A');
    var c=getValue('nombre');
    }
```

Si on ne souhaite pas utiliser l'instruction `with`, il est aussi possible de déclarer une variable et de l'employer de la façon suivante :

```
var o=ggbApplet;
var a=o.getXcoord('A');
var b=o.getYcoord('A');
var c=o.getValue('nombre');
```

Les exemples précédents montrent, au passage, qu'il convient d'écrire entre guillemets les paramètres des méthodes faisant référence à des noms d'objets de la figure. JavaScript permet d'employer indifféremment des guillemets doubles (`"..."`) ou des guillemets simples (`'...'`). On pourra ainsi écrire `getXcoord('A')` ou bien `getXcoord("A")`.

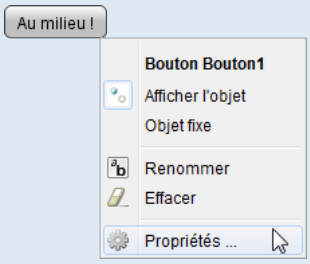
Si on a besoin d'utiliser une apostrophe dans une chaîne de caractères délimitée par des guillemets simples, on fera précéder l'apostrophe du caractère `\`. Par exemple, on pourra écrire : `setCoords('A\'',1,2)` pour affecter les coordonnées $(1;2)$ au point A'.

De la même façon, il est possible d'insérer un guillemet double dans une chaîne de caractères, elle-même encadrée par des guillemets doubles, à condition de l'écrire `\"` au lieu de `"`.

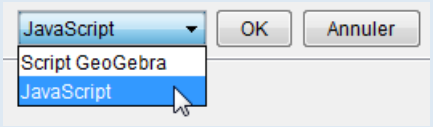
En pratique, pour insérer un script JavaScript dans la figure, la méthode reste très similaire à celle permettant d'insérer un script GeoGebra. Reprenons la situation décrite dans le paragraphe dédié à GeoGebraScript, où il s'agit, à l'appui sur un bouton, d'affecter un point A, libre sur un segment $[MN]$, au milieu de celui-ci.

Méthode

- Cliquer sur l'icône OK.
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la boîte de dialogue ***Bouton***.
- Compléter le champ ***Légende*** avec le texte de votre choix et valider en cliquant sur le bouton Appliquer.
- Effectuer un clic avec le bouton droit de la souris sur le bouton nouvellement créé pour faire apparaître le menu contextuel, et choisir Propriétés...

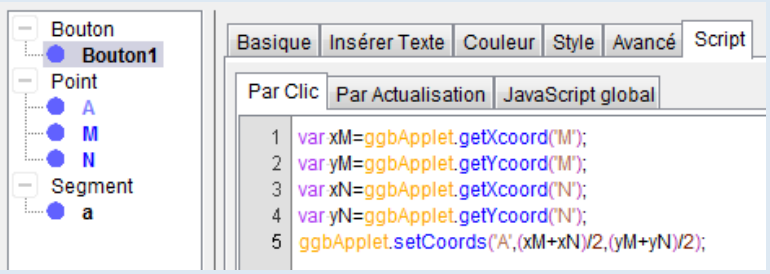


- Sélectionner l'onglet ***Script***.
- Dans l'onglet ***Par Clic***, sélectionner `JavaScript` dans la liste déroulante.



- Taper alors le script suivant :

```
var xM=ggbApplet.getXcoord('M');
var yM=ggbApplet.getYcoord('M');
var xN=ggbApplet.getXcoord('N');
var yN=ggbApplet.getYcoord('N');
ggbApplet.setCoords('A',(xM+xN)/2,(yM+yN)/2);
```



- Valider en cliquant sur le bouton OK.

Ouvrir le fichier exemple

Remarque : Il existe une méthode `setValue(<objet>,<valeur>)` qui permet d'affecter une valeur donnée à un objet donné. Malheureusement, cette méthode est beaucoup plus restrictive que la commande correspondante en GeoGebraScript, **`SoitValeur`**. En effet, seules les variables numériques ou booléennes sont sensibles à la méthode `setValue`. En nommant I le milieu de $[MN]$, on aurait pu tenter d'écrire, dans le script affecté au bouton, `setValue('A','I')`, mais cela serait resté sans effet (sans toutefois déclencher de message d'erreur).

Le lecteur attentif à la liste des méthodes disponibles pour l'objet `ggbApplet` aura certainement remarqué la non-concordance entre l'ensemble des commandes, relatives aux scripts, utilisables en GeoGebraScript et les méthodes dédiées à l'objet `ggbApplet`. En effet, certaines commandes, comme par exemple, **`SoitLégende`**, n'ont pas d'équivalent en JavaScript. Heureusement, les développeurs de GeoGebra ont prévu une méthode générique qui permet d'exécuter, en JavaScript, toute commande GeoGebra.

La méthode `evalCommand(<commande>)` permet d'exécuter la commande GeoGebra `<commande>`. Celle-ci doit être écrite entre guillemets (simples ou doubles) et on doit recourir au nom anglais de la commande GeoGebra pour ne pas entraîner de message d'erreur. Ainsi, pour tracer une droite passant par les points A et B, on écrira : `evalCommand('Line[A,B]')`.

Remarque : En utilisant la méthode `evalCommand`, l'exemple précédent aurait également pu être codé de la manière suivante :

```
ggbApplet.evalCommand('SetValue[A,Midpoint[M,N]]');
```

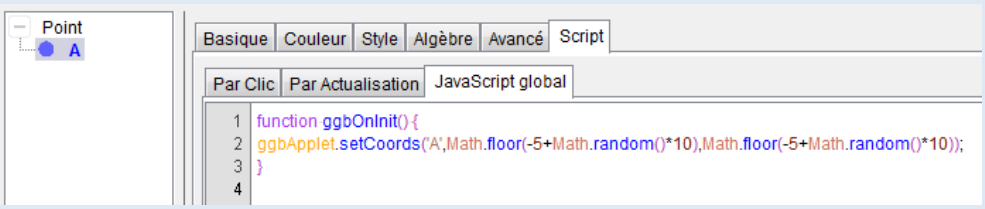
C'est en Javascript qu'il est aussi possible de définir quelques fonctions ou de réaliser quelques affectations globales qui viendront prendre place dans la rubrique ***JavaScript global*** de l'onglet ***Script***. Cette rubrique permet également de définir une fonction particulière, la fonction `ggbOnInit`, qui est automatiquement appelée par GeoGebra une fois que la figure est chargée. Le code situé à l'extérieur de cette fonction est, quant à lui, exécuté avant le chargement de la figure. Il faut donc prendre garde à ne pas utiliser de méthode relative à l'objet `ggbApplet` en dehors de la fonction `ggbOnInit`.

Si on désire, par exemple, qu'un point prenne des coordonnées aléatoires à l'ouverture de la figure, on peut procéder ainsi :

Méthode

- Cliquer sur l'icône  et créer un point A libre dans le plan.
- Effectuer un clic droit sur le point A et, dans le menu contextuel, cliquer sur Propriétés...
- Dans l'onglet ***Script***, rubrique ***JavaScript global***, inscrire :

```
function ggbOnInit() {
    ggbApplet.setCoords('A',Math.floor(-5+Math.random()*10),Math.floor(-5+Math.
    random()*10));
}
```



- Valider en cliquant sur le bouton OK.

`Math` désigne un objet JavaScript qui permet d'accéder à de nombreuses méthodes et propriétés utiles pour effectuer des calculs. En particulier, la méthode `floor` retourne le plus grand entier inférieur ou égal à la valeur qui lui est fournie en paramètre, et la méthode `random()` retourne un nombre aléatoire compris entre 0 et 1.

Dans notre exemple, les coordonnées du point A sont donc entières, et comprises entre -5 et $+5$.

Ouvrir le fichier exemple