Une forme de programmation avec GeoGebra

Sommaire :

Activité 1 : Affichage conditionnel

L'affichage d'un objet dans GeoGebra peut être conditionné par une case à cocher, par un bouton, par un curseur ou par toute autre condition numérique ou géométrique.

Vous trouverez un extrait de la brochure "Créer avec GeoGebra", pages 232 à 236, en annexe 1.

En vous référant à ce document, créer les fichiers suivants :

a) *Avec une case à cocher :*
Créer une figure possédant un axe de symétrie et faire apparaître ou disparaître cet axe de symétrie à l'aide d'une case à cocher.

b) *Avec un bouton :*
Reprendre la figure de l'exercice a) et gérer la visibilité à l'aide d'un bouton.

c) *Avec un curseur :*
Dessiner un sapin (pour plus de rapidité, utiliser l'aimantation et la grille) puis un disque faisant office de boule de Noël. Mettre l'opacité de ce disque à 100% et sa couleur en rouge.
Créer un curseur prenant des valeurs entières entre 0 et 1.
Faire en sorte que la boule soit visible seulement si le curseur vaut 1.
Mettre la vitesse du curseur à 10 et l'animer.

d) *Avec une condition numérique ou géométrique :*
Créer un segment [AB] et un point M.
Créer un point N superposé au point M en tapant dans la barre de saisie : N=M
Mettre la couleur de ce point N en rouge, désactiver l'affichage de l'étiquette et activer sa trace.
A l'aide de la commande Distance[<Point A> , <Point B>], conditionner l'affichage du point N par le fait que la différence entre les distances AM et BM soit inférieure à 0.05. (Pour cela, la commande Abs(<Nombre>) est bien utile).

Activité 2 : Simuler des boutons radio à l'aide de cases à cocher

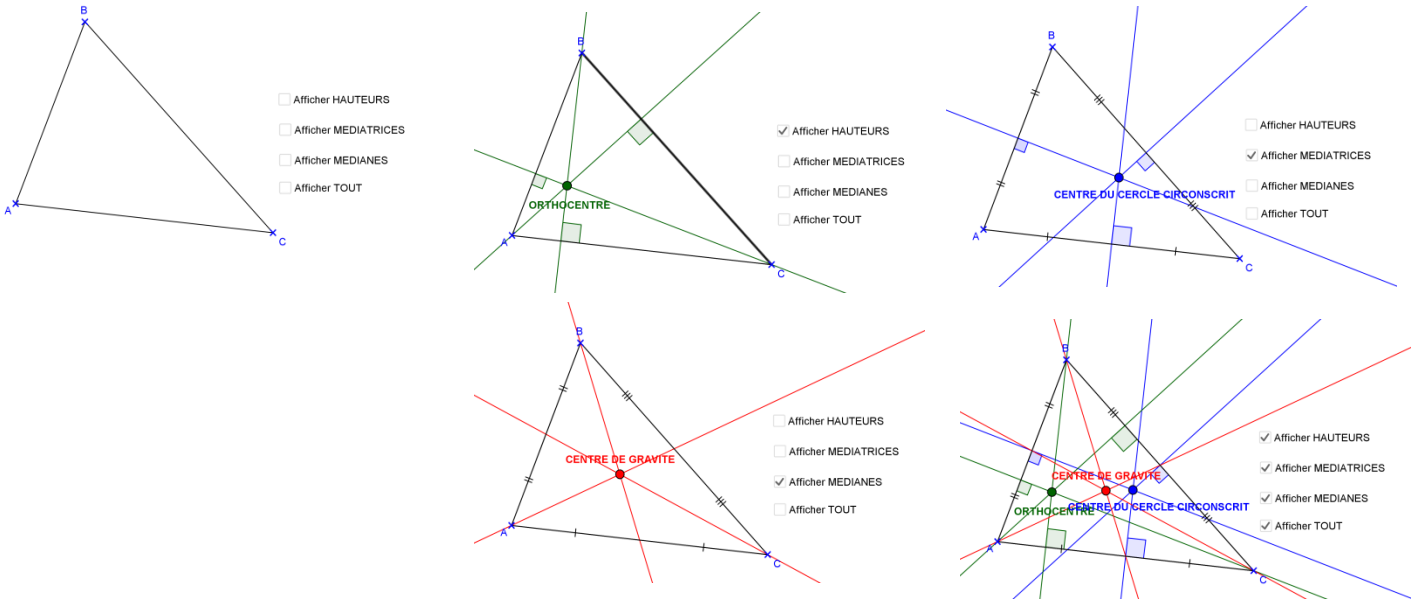
Vous trouverez un extrait de la brochure "Créer avec GeoGebra", pages 381 à 382, en annexe 2.

En vous référant à ce document, créer un fichier GeoGebra dans lequel on tracera un triangle quelconque et on fera apparaître soit les médianes, soit les hauteurs, soit les médiatrices, soit les bissectrices, soit les hauteurs-médianes-médiatrices.

On pourra également ajouter la droite d'Euler.

(pensez à utiliser des couleurs différentes pour augmenter la lisibilité de la figure)

Voici un aperçu du résultat attendu :



Activité 3 : Utiliser les boutons

Cliquez sur l'icône 

.

Cliquez à l'endroit de la feuille où vous voulez placer le bouton, notez dans la légende le texte à écrire sur le bouton puis écrire le script correspondant à l'action désirée.

Exemples de scripts :

DémarrerAnimation[ ]

Reprend toutes les animations qui sont en "Pause".

DémarrerAnimation[<Booléen b>]

Si *b*=false, met en "Pause" toutes les animations, sinon les reprend.

DémarrerAnimation[<curseur a>, <false|true>]

Reprend l'animation du curseur a (qui doit être en pause) si le deuxième argument est true, met en "Pause" l'animation du curseur a si le deuxième argument est false.

SoitValeur[<Booléen b>, <0|1>]

Affecte au booléen *b* la valeur 0 ou 1.
Vous pouvez aussi utiliser SoitValeur[<Booléen b>, <false|true>]

SoitValeur[<Objet A>, <Objet B>]

Si *A* est un objet libre ou un point appartenant à un chemin ou une région, sa valeur prend la valeur actuelle de *B*, créé ou non (i.e. *A* ne changera pas de valeur si *B* est ensuite modifié).

SoitValeur[<Liste L>, <Nombre n >, <Objet O>]

Affecte au $n^{ième}$ élément de la liste *L* la valeur actuelle de l'objet O .
Le nombre *n* est au plus égal à Longueur[L]+1.

SoitRemplissage[<Objet>, <Nombre n>]

Modifie l'opacité de l'objet. Le nombre *n* appartient à l'intervalle [0,1], où 0 signifie 'transparent' et 1 '100% opaque'.
Les autres valeurs sont ignorées.

ActualiserConstruction[]

Recalcul de tous les objets utilisés dans le fichier GeoGebra.

Agrandir[<Facteur>]

Agrandit le Graphique avec le facteur donné, le centre de l'écran étant utilisé comme centre d'agrandissement.
Astuce : Agrandir[1] ne fait *rien* (mais cela provoque la disparition des traces d'objets) ;

Agrandir[<Facteur>, <Centre>]

Agrandit le Graphique avec le facteur donné, le second paramètre précisant le centre d'agrandissement.

Agrandir[<Min x>, <Min y>, <Max x>, <Max y>]

Agrandit le graphique au rectangle construit sur les sommets (Min x, Min y), (Max x, Max y).

SoitVisibleDansVue[<Objet>, <Numéro 1|2|-1>, <Booléen>]

Rend l'objet visible ou non dans Graphique 1 ou 2 ou Graphique 3D (-1).

Si on veut créer un script qui agit comme interrupteur d'animation du point M, on peut créer le booléen a et lui donner, par exemple, la valeur "false" puis utiliser le modèle suivant :

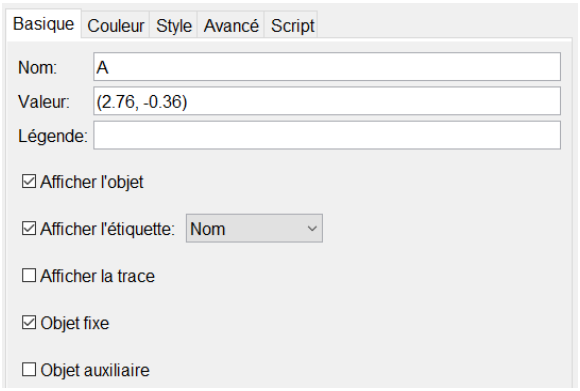
"SoitValeur(a,!a)
DémarrerAnimation[M, a]"

Le symbole "!" veut dire "non".
"!a" renvoie donc "true" si a valait "false" et "false" si a valait "true".

Exercice d'application :

Création de la figure de base :

Tracer un triangle ABC puis fixer les points A et B. (clic droit / Propriétés / Objet fixe)



Marquer l'angle $\widehat{ACB}$ sans afficher sa mesure.

Création du bouton permettant de copier le point C quand on pense que le triangle est rectangle :

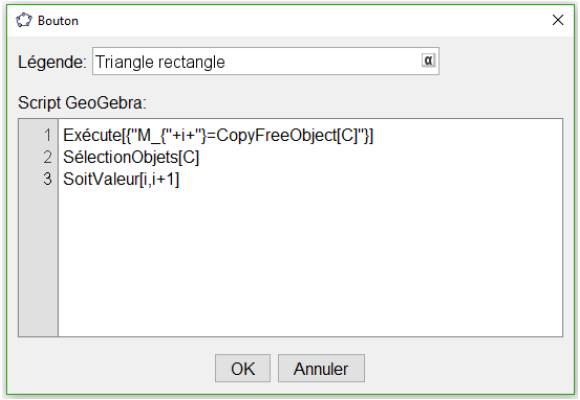
Créer un nombre i égal à 1 : "i=1"

Créer un bouton « Triangle rectangle » avec pour script :

Exécute[{"M_{"+i+"}=CopyFreeObject[C]"}]

SélectionObjets[C]

SoitValeur[i,i+1]



La commande *Exécute* permet d'exécuter une liste de textes qui contiennent des commandes comme si on les écrivait dans le champ de saisie, sauf qu'il faut qu'ils soient... en anglais !

Ici, on aurait pu se contenter de la commande *CopierObjetLibre[C]* mais elle ne permettrait pas de gérer les noms attribués à chaque point et on en aura besoin quand on voudra réinitialiser la figure.

La commande *SélectionObjet* permet de sélectionner le point C après l'avoir cloné. Si on n'écrit pas cette commande, lorsqu'on voudra déplacer le point C, c'est son clone qui se déplacera puisque ce sera le dernier objet créé.

La commande *SoitValeur[i,i+1]* permet d'incrémenter le compteur pour que l'activité se

termine au bout de 20 points.

Création du bouton permettant la visualisation du cercle solution :

Créer le point L milieu du segment [AB] et le cercle l de diamètre [AB].

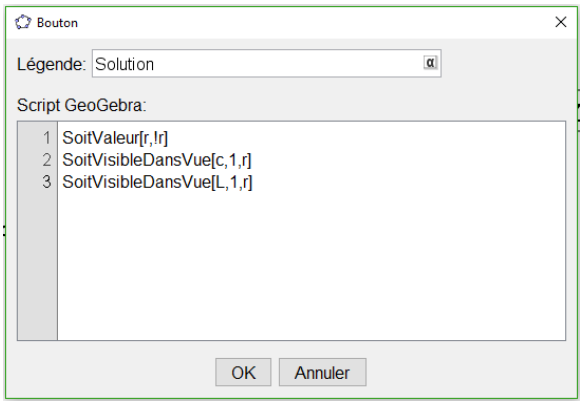
Créer le booléen r qu'on initialisera à *false* (ou à 0) : *r=false*

Créer le bouton « Solution» dans le script duquel on note :

SoitValeur[r,!r]

SoitVisibleDansVue[c,1,r]

SoitVisibleDansVue[L,1,r]



La commande *SoitValeur[r,!r]* permet de remplacer la valeur du booléen *r* par son contraire.
La commande *SoitVisibleDansVue[<Objet>,<numéro 1|2|-1>,<Booléen>]* permet de rendre visible l'objet <Objet> dans la vue graphique (1), graphique 2 (2) ou graphique 3D (-1) si le booléen <Booléen> est vrai et de le rendre invisible sinon.

Dans l'onglet "Avancé" des propriétés de ce bouton, noter la condition d'affichage $i \geq 20$.

Création du bouton permettant de réinitialiser la figure :

Créer un bouton "RAZ" dont le script est :

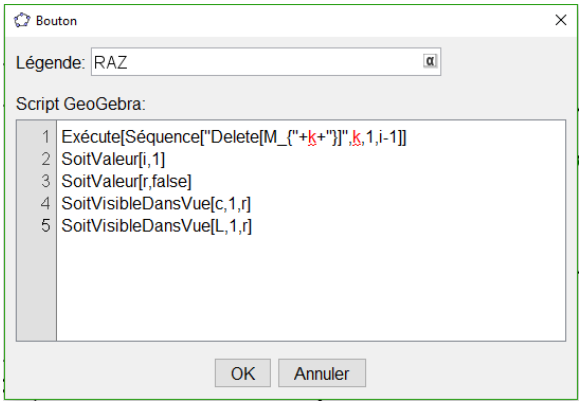
Exécute[Séquence["Delete[M_{"+k+"}]",k,1,i-1]]

SoitValeur[i,1]

SoitValeur[r,false]

SoitVisibleDansVue[c,1,r]

SoitVisibleDansVue[L,1,r]



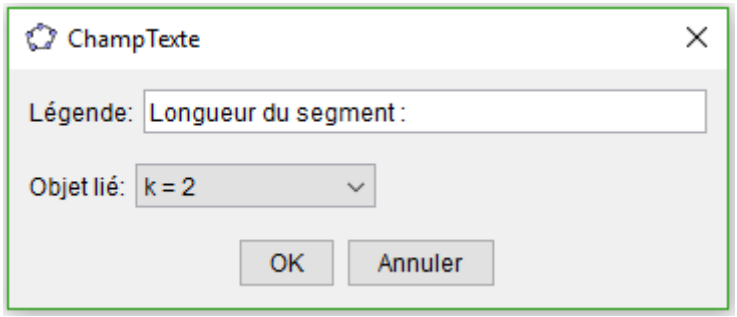
La commande *Exécute[Séquence["Delete[M_{"+k+"}]",k,1,i-1]]* permet d'effacer les points M_k tracés, avec k allant de 1 à i-1.

Activité 4 : Utiliser les ChampTextes

Un ChampTexte permet à l'utilisateur de taper une valeur au clavier pour l'attribuer à une variable déjà définie.
Par exemple, si on veut que l'utilisateur puisse modifier la longueur d'un segment :
Créer un point A (première extrémité du segment).
Créer une variable k (longueur du segment) et lui attribuer une valeur de votre choix.
Créer ensuite le segment d'extrémité A et de longueur k (outil "segment de longueur donnée").

Créer enfin un ChampTexte à l'aide de l'outil 

:



Vous pouvez réduire la taille de la fenêtre associée au ChampTexte dans les propriétés de ce dernier.

Exemples d'application :

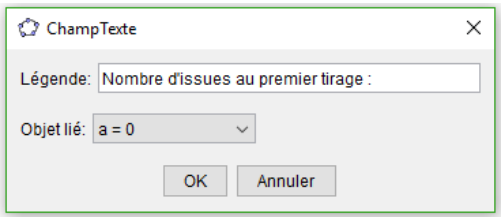
a) Arbres de probabilité :

Placer les points suivants : A=(-2,0), F=(9,6), G=(9,-6), H=(20,6), I=(20,-6)

Créer deux variables $a = 0$ et $b = 0$.

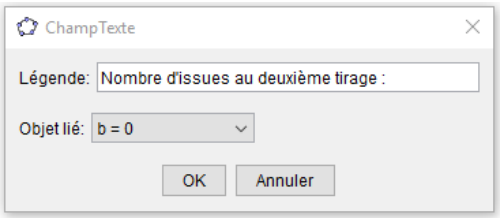
Créer un ChampTexte à l'aide de l'icône 

et renseigner comme ci-dessous :



Ce ChampTexte permettra de changer la valeur de la variable a en tapant au clavier la nouvelle valeur.

Créer un deuxième ChampTexte et renseigner comme ci-dessous :



On peut réduire la longueur de ces Champtextes en sélectionnant l'onglet style de leurs propriétés.

Créer ensuite une liste de a points d'abscisse 9 :
$liste1 = Séquence[F + i * (G - F)/(a + 1), i, 1, a]$.
Les a points créés sont régulièrement espacés entre F et G.

Créer maintenant une liste de $a \times b$ points d'abscisse 20 :
$liste2 = Séquence[H + i\ (I - H)\ /\ (a * b + 1), i, 1, a * b]$
Les $a * b$ points créés sont régulièrement espacés entre H et I.

Créer une liste de segments reliant le point A aux a points alignés avec F et G :
$ListeSegments1 = Séquence[Segment[A, Elément[liste1, i]], i, 1, a]$

Créer une liste de segments reliant chaque point de la liste $liste1$ aux b points correspondants de la liste $liste2$. Pour cela, il faut créer une liste de listes de segments :
$ListeSegments2 = Séquence[Séquence[Segment[Elément[liste1, i], Elément[liste2, b\ (i - 1) + j]], j, 1, b], i, 1, a]$

On peut maintenant "nettoyer" le document en cachant les points A, F, G, H et I ainsi que les listes $liste1$ et $liste2$. On peut également changer la couleur des segments de la $ListeSegments1$ ainsi que celle des segments de la liste $ListeSegments2$.

b) Cercle/disque à paramétrer :

- Créer les variables suivantes : x_A, y_A, r, a, R, V et B (leur donner une valeur de votre choix)
- Créer le point A=(x_A,y_A).
- Créer le cercle c de centre A et de rayon r.
- Créer un ChampTexte nommé "Abscisse de A" qui permet de modifier x_A.
- Créer un ChampTexte nommé "Ordonnée de A" qui permet de modifier y_A.
- Créer un ChampTexte nommé "Rayon" qui permet de modifier r.
- Créer un ChampTexte nommé "Remplissage du disque entre 0 et 1" qui permet de modifier a.
  puis, dans le script de ce ChampTexte, utiliser la commande SoitRemplissage[<Objet>,<Nombre>] pour modifier l'opacité du remplissage du disque c à l'aide du nombre a.
- Créer un ChampTexte nommé " Couleur rouge du disque entre 0 et 1" qui permet de modifier R.
  puis, dans le script de ce ChampTexte, utiliser la commande SoitCouleur[<Objet>,<Rouge>,<Vert>,<Bleu>] pour que la couleur du disque c soit gérée par les couleurs R, V et B.
- Créer un ChampTexte nommé "Couleur verte du disque entre 0 et 1" qui permet de modifier V.
  puis, dans le script de ce ChampTexte, utiliser la commande SoitCouleur[<Objet>,<Rouge>,<Vert>,<Bleu>] pour que la couleur du disque c soit gérée par les couleurs R, V et B.
- Créer un ChampTexte nommé " Couleur bleue du disque entre 0 et 1" qui permet de modifier B.
  puis, dans le script de ce ChampTexte, utiliser la commande SoitCouleur[<Objet>,<Rouge>,<Vert>,<Bleu>] pour que la couleur du disque c soit gérée par les couleurs R, V et B.

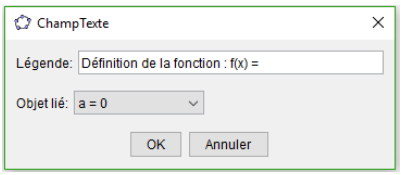
Activité 5 : Utiliser les couleurs conditionnelles

L'objectif de ce document est de créer un fichier qui mette en évidence la relation entre les variations d'une fonction et le signe de sa dérivée. (Voir un aperçu du document final en ouvrant le fichier "Variations d'une fonction.ggb")

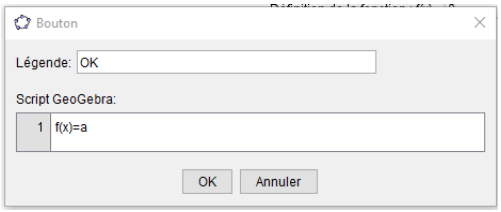
Faire apparaître les axes du repère.

Créer une variable a=0

Créer un ChampTexte nommé "Définition de la fonction : f(x) = ", lié à l'objet a.



Créer un bouton nommé "OK" avec le script : f(x)=a.



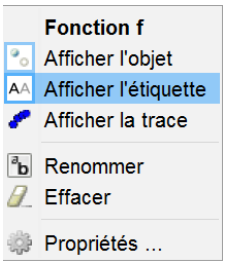
Dans les propriétés de ce bouton, changer sa taille comme indiqué ci-contre :

Positionner ce ChampTexte et ce bouton l'un à côté de l'autre

.

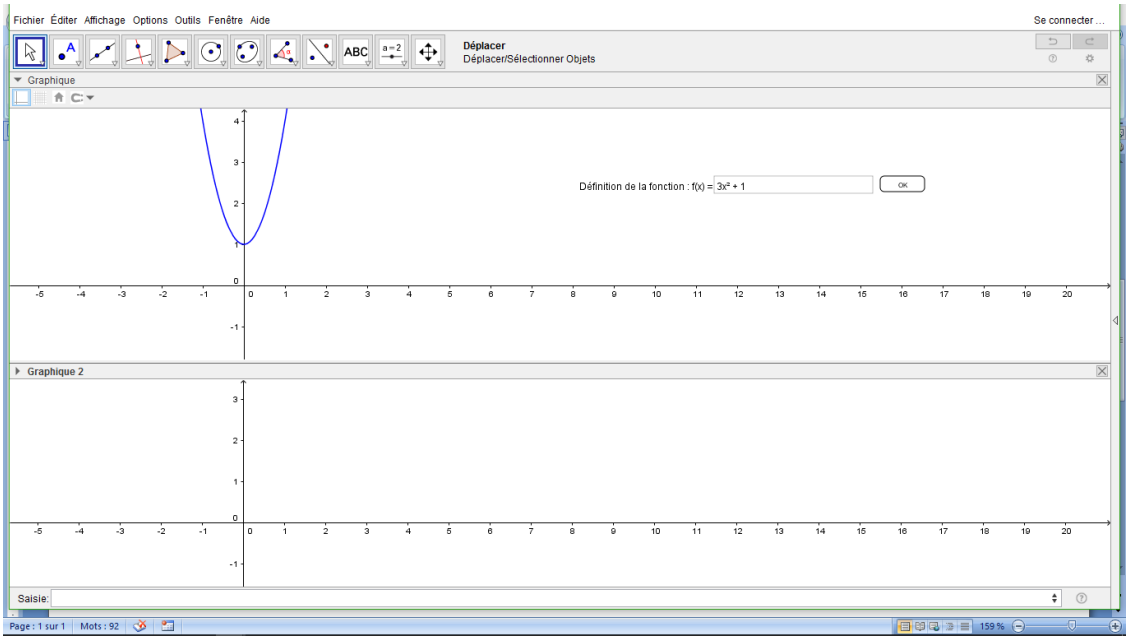
Tester ces objets en complétant le ChampTexte et en appuyant sur le bouton.

Si ce n'est pas le cas, afficher l'étiquette de la fonction ainsi tracée.

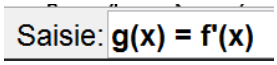


Afficher la vue Graphique 2 et faire apparaître les axes du repère.

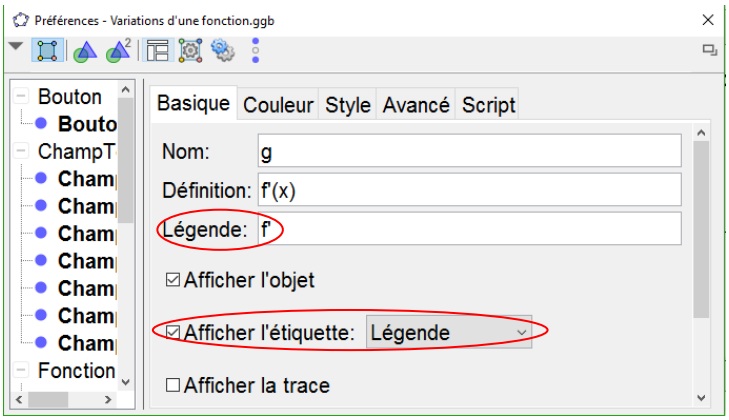
Changer la disposition des vues de façon à obtenir le visuel ci-dessous :



Cliquer dans la vue Graphique 2 pour travailler dans celle-ci.

Créer la fonction g(x)=f'(x). 

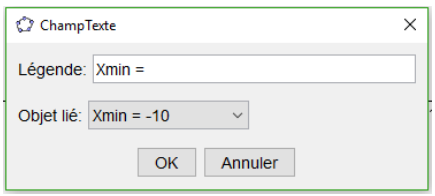
Dans les propriétés de cette fonction, faire les réglages suivants :



On va ajuster la taille des vues Graphique et Graphique 2 aux courbes représentant les fonctions f et f' :

Créer six variables Xmin=-10 et Xmax=10, Ymin_1=-2 et Ymax_1=3, Ymin_2=-2 et Ymax_2=3.

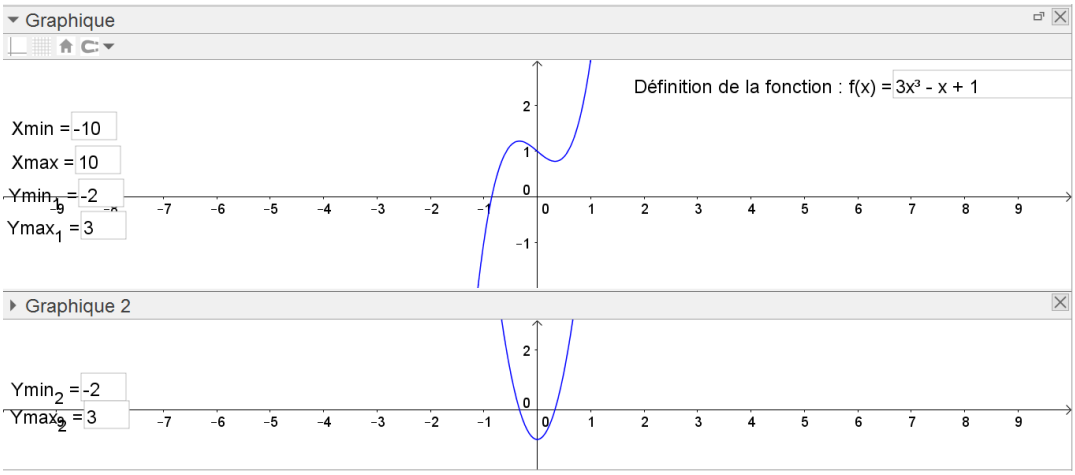
Dans la vue Graphique, créer quatre ChampTextes "Xmin = ", "Xmax = ", "Ymin_1 = " et "Ymax_1 = " permettant de modifier ces valeurs.



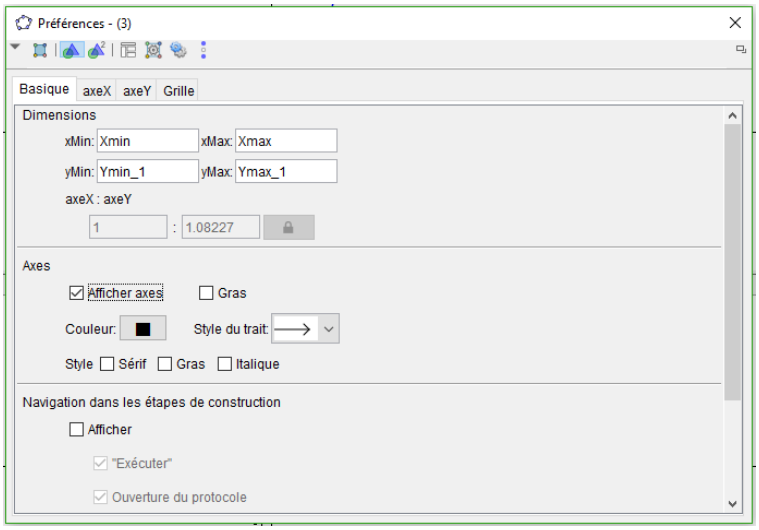
Dans la vue Graphique 2, créer deux ChampTextes "Ymin_2 = " et "Ymax_2 = " permettant de modifier ces valeurs.

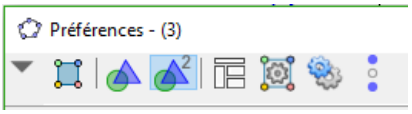
Diminuer la longueur de ces six ChampTextes à la valeur 3 :

Placer ces ChampTextes sur la gauche de l'écran, l'un en dessous de l'autre.



A l'aide d'un clic droit dans une zone vierge de la vue Graphique, sélectionner Graphique... et modifier les dimensions comme ci-dessous :



Dans cette même fenêtre, sélectionner la vue Graphique 2 

puis régler les dimensions de cette dernière, de façon analogue.

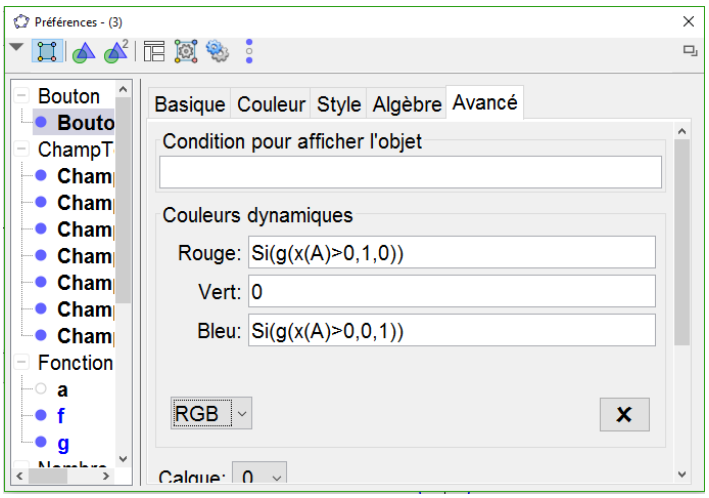
On a maintenant deux repères, l'un au dessus de l'autre avec des axes des abscisses qui correspondent.

Placer un point A sur la courbe représentative de la fonction f, sur la gauche de la vue Graphique.

Construire un point A' sur la courbe représentative de la fonction g, sur la vue Graphique 2, ayant la même abscisse que le point A : Saisie: **A' = (x(A), g(x(A)))**

Enlever l'affichage de l'étiquette de chacun de ces deux points.

A l'aide de la touche Ctrl, sélectionner ces deux points et, dans leurs propriétés,
- activer la trace
- choisir le même style,
- dans l'onglet "avancé", renseigner les couleurs comme ci-dessous :



De cette façon, les deux points seront rouges si le nombre dérivé de la fonction f en x(A) est positif et bleus sinon.

Dans l'onglet Algèbre des propriétés du point A, vérifier que le réglage du champ "Répéter" soit bien sur ⇒ Croissant :

A l'aide d'un clic droit sur le point A, animer ce point.

Il ne reste plus qu'à renseigner les ChampTextes avec les données choisies pour voir apparaître les relations entre le signe du nombre dérivé et les variations de la fonction f.

Remarque : Ce document permet également d'illustrer, entre autres, le fait que l'ajout d'un réel à la définition d'une fonction ne modifie pas sa dérivée !

Fiche technique

21 Montrer ou cacher un objet

http://url.univ-irem.fr/ft8

QR Code

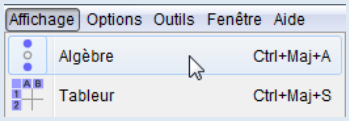
Considérons un point A que l'on cherche à rendre visible ou bien invisible.

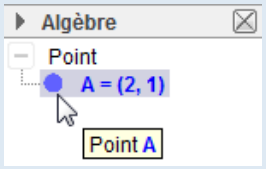
1 Manuellement

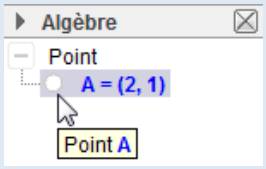
L'interface de GeoGebra offre plusieurs possibilités pour montrer ou cacher un objet.

Méthode

- Si la vue ***Algèbre*** n'est pas présente à l'écran, la faire apparaître à l'aide de la commande Affichage ▶ ⋮ Algèbre .
- Dans la vue ***Algèbre*** cliquer sur le disque coloré ● situé devant le nom du point A.





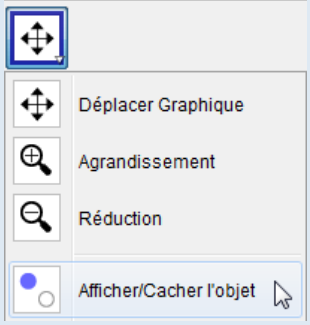


- Le point A n'est alors plus visible dans la fenêtre du graphique. Pour le faire réapparaître, il suffit de cliquer à nouveau sur le disque (vide cette fois-ci) ○ situé face au nom du point.

L'appui sur l'icône  permet de sélectionner les objets à cacher ou à montrer.

Méthode

- Cliquer sur l'icône  .
- Sélectionner le point A avec le bouton gauche de la souris.
- Changer de mode en cliquant, par exemple, sur le bouton  .



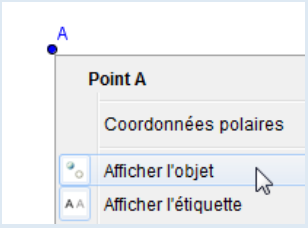
Remarque :

Ce bouton se révèle fort pratique à l'usage lorsqu'il s'agit de rendre visibles des objets auparavant cachés. En effet, l'appui sur  provoque l'affichage temporaire des objets cachés qu'il suffit alors de sélectionner (on peut en choisir plusieurs à la fois à l'aide de la touche Ctrl) pour changer leur statut (ne pas oublier de basculer sur un autre mode pour rendre la modification effective).

Le menu contextuel permet également de rendre invisible un objet.

Méthode

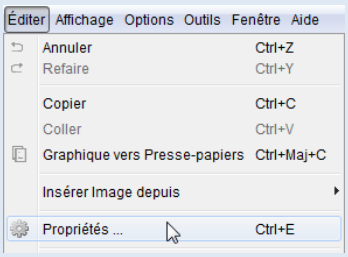
- Faire apparaître le menu contextuel en cliquant avec le bouton droit de la souris sur un objet.
- Cocher ou décocher l'item Afficher l'objet .

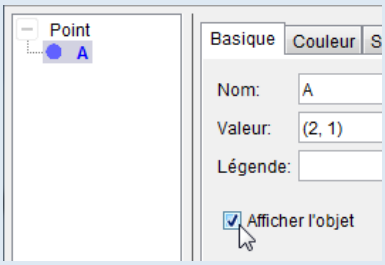


Le panneau des propriétés des objets offre également l'accès aux modifications de visibilité.

Méthode

- Ouvrir la fenêtre ***Préférences*** en passant par le menu Éditer ▶ Propriétés... .
  S'il s'agit de cacher un objet, il est également possible de faire apparaître le menu contextuel en effectuant un clic droit sur l'objet visible puis de choisir Propriétés... .
- Dans la rubrique de gauche, sélectionner un (ou plusieurs) objet(s).
- Dans l'onglet ***Basique***, décocher ***Afficher l'objet***.



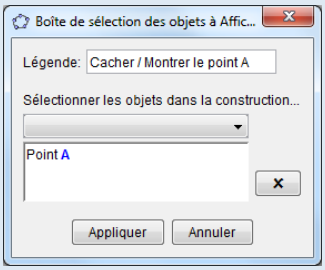


2 Avec une case à cocher

L'utilisation de cases à cocher au sein d'un imagiciel permet d'afficher ou de cacher rapidement plusieurs objets à la fois.

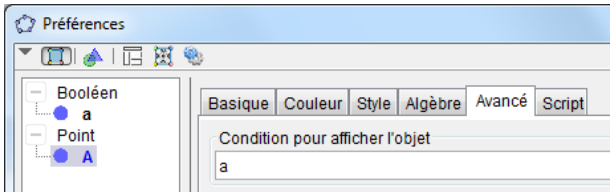
Méthode

- Cliquer sur l'icône  pour créer une case à cocher.
- Cliquer sur une zone vierge de vue ***Graphique*** pour provoquer l'apparition de la fenêtre ***Boîte de sélection des objets à Afficher/Cacher***.
- Compléter le champ ***Légende*** avec le texte voulu.
- Dans la liste déroulante, sélectionner les objets dont la visibilité doit dépendre de l'état de la case à cocher.



- Valider en cliquant sur le bouton Appliquer.

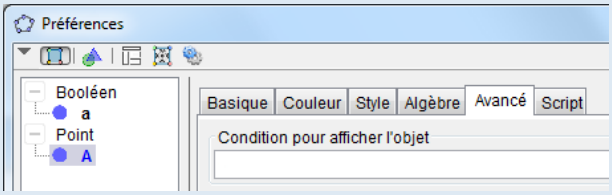
L'insertion d'une case à cocher entraîne, d'une part, la création d'un booléen (visible dans la fenêtre ***Algèbre*** et dont le nom est choisi par GeoGebra, mais il est évidemment possible de le renommer à posteriori) qui prend la valeur `true` lorsque la case est cochée ou la valeur `false` sinon. D'autre part, chacun des objets sélectionnés lors de la création de la case à cocher voit sa visibilité liée à la valeur du booléen. Pour ce faire, le champ ***Condition pour afficher l'objet*** (onglet ***Avancé*** dans les propriétés des objets) est automatiquement rempli par GeoGebra avec le nom du booléen.

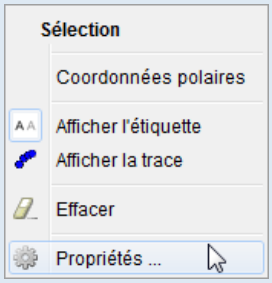


La fenêtre ***Boîte de sélection des objets à Afficher/Cacher*** n'apparaît qu'au moment de la création de la case à cocher. Si on souhaite ajouter ou supprimer des objets dans la liste des objets dont la visibilité dépend de l'état d'une case à cocher, il convient alors de modifier le champ ***Condition pour afficher l'objet*** dans les propriétés de ces objets.

Méthode

- Faire apparaître le menu contextuel d'un objet en effectuant un clic droit sur celui-ci et choisir Propriétés... .
- Dans l'onglet ***Avancé***, supprimer ou ajouter le nom du booléen dans le champ ***Condition pour afficher l'objet***.





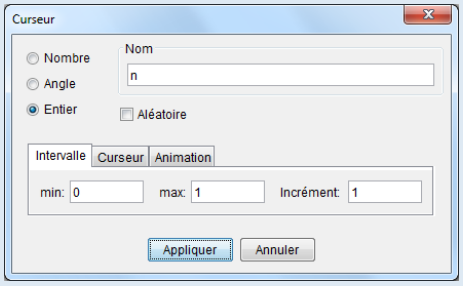
Ouvrir le fichier exemple

3 Avec un curseur

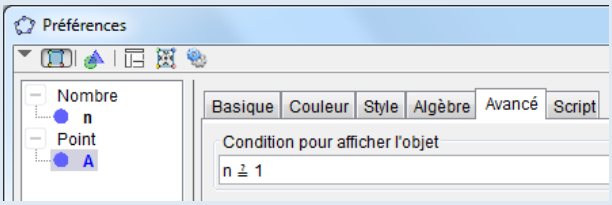
Comme il a été vu dans le paragraphe précédent, le champ ***Condition pour afficher l'objet*** permet d'agir sur la visibilité d'un objet en fonction d'une condition, ce qui peut se révéler très intéressant si on désire mettre en œuvre une condition plus sophistiquée qu'un simple `true`/`false` obtenu à l'aide d'une case à cocher.

Méthode

- Cliquer sur l'icône  .
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la boîte de dialogue ***Curseur***.
- Sélectionner ***Entier*** et choisir 0 pour borne inférieure et 1 pour borne supérieure (on laissera l'incrément à 1).



- Valider en cliquant sur le bouton  .
- Faire apparaître le panneau des propriétés du point A.
- Dans l'onglet ***Avancé***, compléter le champ ***Condition pour afficher l'objet*** avec la condition suivante : `n==1`.



Dès lors, le mouvement du curseur entraîne l'apparition ou la disparition du point A.

Le lecteur avisé objectera qu'une telle façon de procéder ne présente aucun avantage par rapport à l'utilisation d'une case à cocher. Et il aura parfaitement raison, puisqu'on se contente ici de remplacer un `true`/`false` par un `0`/`1`! Néanmoins, la méthode décrite ici peut être généralisée à un curseur dont la borne supérieure est plus grande que 1 et permet, en particulier, d'afficher les étapes de construction d'une figure les unes après les autres. La condition de visibilité du premier objet à afficher (ou du premier groupe d'objets) sera alors `n>0`, celle du second objet (ou groupe d'objets) `n>1`, ... , et celle du dernier objet `n==<borne supérieure du curseur>` (les tests et opérateurs logiques sont traités dans la fiche technique **Les valeurs booléennes**, page 593).

Ouvrir le fichier exemple

4 Avec un bouton

La modification de la visibilité d'un objet à l'aide d'un bouton donne l'occasion de pratiquer les langages de scripts intégrés à GeoGebra.

Méthode

- Définir un booléen nommé, par exemple, `visibleA` ayant pour valeur `true`. Pour cela, inscrire dans la barre de saisie : `visibleA=true` et valider en appuyant sur ↵.

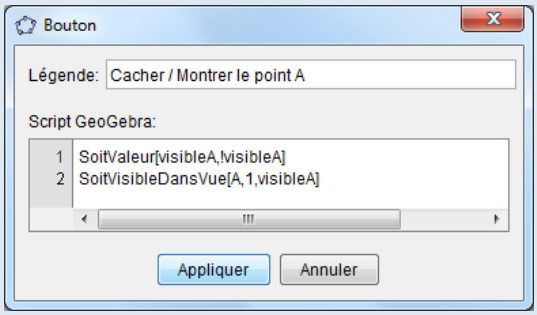
- Cliquer sur l'icône  .
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la

boîte de dialogue ***Bouton***.

- Compléter le champ ***Légende***.
- Dans la rubrique ***Script GeoGebra***, inscrire :

```
SoitValeur[visibleA,!visibleA]
SoitVisibleDansVue[A,1,visibleA]
```

- Valider en cliquant sur le bouton Appliquer.



L'instruction `SoitValeur[<booléen>,<0|1>]` permet d'affecter la valeur 0 (`false`) ou 1 (`true`) au booléen `<booléen>`.

La syntaxe `!<booléen>` désigne le contraire de `<booléen>`, c'est-à-dire, non-`<booléen>` (voir la fiche technique **Les valeurs booléennes**, page 593).

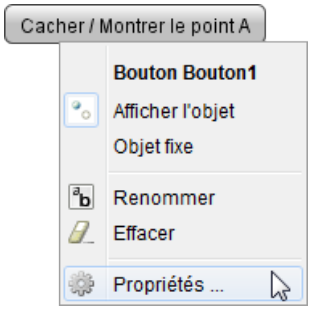
La commande `SoitValeur[visibleA,!visibleA]` agit donc comme une bascule pour modifier la valeur de la variable booléenne `visibleA` (passage de `true` à `false` ou de `false` à `true`).

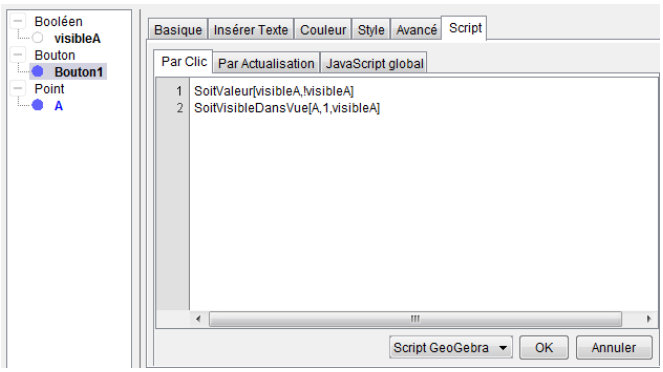
L'instruction `SoitVisibleDansVue[<objet>,<numéro 1|2>,<booléen>]` permet d'afficher (si `<booléen>` vaut `true`) ou de cacher (si `<booléen>` vaut `false`) l'objet `<objet>` dans la fenêtre de graphique 1 ou 2.

La commande `SoitVisibleDansVue[A,1,visibleA]` permet ainsi de rendre dépendante du booléen `visibleA` la visibilité du point A (en supposant que le point A a été construit dans la fenêtre de graphique n° 1).

Remarque :

- Lorsque, pour une raison quelconque, on désire, comme dans l'exemple ci-dessus, afficher ou cacher un objet à l'aide d'un bouton, il paraît également judicieux de modifier dynamiquement la légende du bouton en fonction de l'état de l'objet. Pour cela, on se reportera à la fiche **Rendre dynamique la légende d'une case à cocher (ou d'un bouton, ou ...)** (page 723).
- Pour modifier le script attaché à un bouton, faire apparaître le panneau des propriétés du bouton et, dans l'onglet ***Script***, sélectionner l'onglet ***Par clic***.





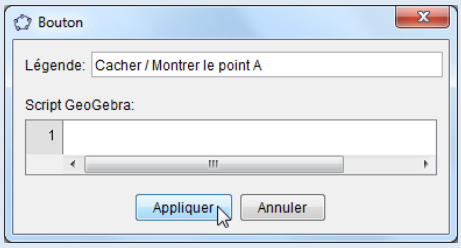
Une fois les modifications effectuées, ne pas oublier de les valider en cliquant sur le bouton OK

Ouvrir le fichier exemple

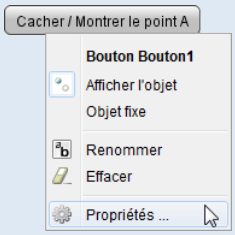
JavaScript vous permet aussi de modifier la visibilité d'un objet à volonté.

Méthode

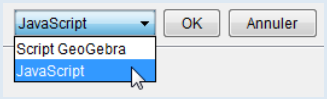
- Cliquer sur l'icône OK.
- Cliquer sur une zone vierge de la fenêtre de graphique pour provoquer l'apparition de la boîte de dialogue ***Bouton***.
- Compléter le champ ***Légende***.



- Sans rien inscrire dans la rubrique ***Script GeoGebra***, valider en cliquant sur le bouton Appliquer.

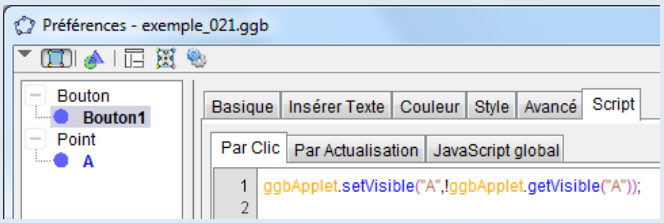


- Effectuer un clic droit sur le bouton ainsi créé pour faire apparaître le menu contextuel et choisir Propriétés... .
- Choisir l'onglet ***Script*** ;
- Dans l'onglet ***Par Clic***, sélectionner `JavaScript` dans la liste déroulante.



- Puis, taper la commande :

```
ggbApplet.setVisible("A",!ggbApplet.getVisible("A"));
```



- Valider en cliquant sur le bouton OK.

L'instruction `setVisible(<objet>,<booléen>)` permet de rendre visible (si `<booléen>` vaut `true`) ou invisible (si `<booléen>` vaut `false`) l'objet `<objet>`, quelle que soit la fenêtre dans laquelle `<objet>` est construit.

L'instruction `getVisible(<objet>)` renvoie un booléen : `true` si `<objet>` est visible, `false` sinon.

Le point d'exclamation `!` devant une variable booléenne désigne l'opérateur de négation.

Ainsi, la commande `ggbApplet.setVisible("A",!ggbApplet.getVisible("A"))` rend le point A visible s'il était auparavant invisible ou A invisible s'il était visible.

Ouvrir le fichier exemple

Fiche technique 36

Simuler des boutons radio à l'aide de cases à cocher

http://url.univ-irem.fr/ft4

QR Code

Des boutons radio sont des boutons qui fonctionnent ensemble : quand l'un est coché, les autres ne le sont pas. Bien qu'il ne soit pas possible d'insérer directement des boutons radio dans GeoGebra, il est malgré tout possible d'en simuler le fonctionnement en se servant des cases à cocher.

☐ Case a ☐ Case b ☑ Case c

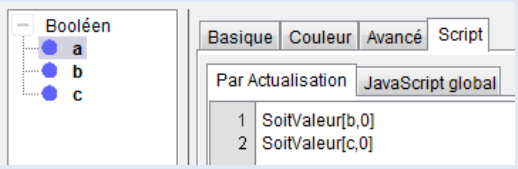
Dans la suite, nous appellerons a, b et c les booléens associés aux trois cases à cocher.

1 En Script GeoGebra

Méthode

- Ouvrir la boîte des propriétés de la case à cocher nommée a.
- Choisir l'onglet ***Script***.
- Dans l'onglet ***Par actualisation***, inscrire les commandes :

```
SoitValeur[b,0]
SoitValeur[c,0]
```



- Valider en cliquant sur le bouton OK.
- Recommencer la procédure pour les cases b et c en modifiant les noms des booléens dans la zone de script de la façon suivante :

Case b

```
SoitValeur[a,0]
SoitValeur[c,0]
```

Case c

```
SoitValeur[a,0]
SoitValeur[b,0]
```

L'instruction `SoitValeur[<objet>,<valeur>]` affecte la valeur `<valeur>` à l'objet `<objet>`.

Dans notre exemple, les objets étant des booléens, on peut leur affecter les valeurs 0 (`false`) ou 1 (`true`). Pour simuler le comportement de boutons radio, l'idée consiste à affecter la valeur 0 aux deux booléens qui

correspondent aux cases qui ne sont pas cochées (ou décochées) par l'utilisateur.

Remarque :

Lorsqu'il s'agit de véritables boutons radio, au moins l'un des boutons est nécessairement actif. Si l'on désire imiter ce comportement, il suffit d'affecter systématiquement la valeur 1 à la case modifiée par l'utilisateur. Ainsi, le script associé à la case a deviendrait :

```
SoitValeur[a,1]
SoitValeur[b,0]
SoitValeur[c,0]
```

Évidemment, il convient de modifier de la même façon les scripts associés aux cases b et c.

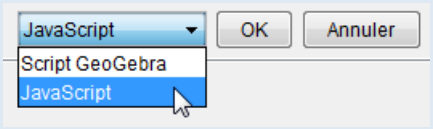
Ouvrir le fichier exemple

2 En JavaScript

Le principe expliqué au paragraphe précédent reste valable en JavaScript, seule la syntaxe change.

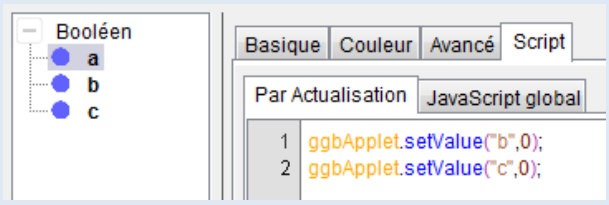
Méthode

- Ouvrir la boîte des propriétés de la case à cocher nommée a.
- Choisir l'onglet ***Script***.
- Dans l'onglet ***Par actualisation***, sélectionner `JavaScript` dans la liste déroulante.



- Puis, taper les commandes :

```
ggbApplet.setValue("b",0);
ggbApplet.setValue("c",0);
```



- Valider en cliquant sur le bouton OK.
- Recommencer la procédure pour les cases b et c en modifiant les noms des booléens dans la zone de script de la façon suivante :

Case b

```
ggbApplet.setValue("a",0);
ggbApplet.setValue("c",0);
```

Case c

```
ggbApplet.setValue("a",0);
ggbApplet.setValue("b",0);
```

L'instruction `setValue[<objet>,<valeur>]` affecte la valeur `<valeur>` à l'objet `<objet>`.

Remarque :

En JavaScript, les objets GeoGebra doivent être désignés par des chaînes de caractères et de ce fait, le nom des objets doit être encadré par des guillemets simples (`'...'`) ou par des guillemets doubles (`"..."`).

Ouvrir le fichier exemple