Createxo --- Aide ---

Le format OEF

OEF veut dire ``online exercise format'', un format pour des exercices mathématiques destinés aux systèmes d'enseignement assisté par ordinateur. Le but de la création de ce format est de favoriser les échanges de contenu entre différents systèmes. Ce format contient donc uniquement des informations mathématiques de l'exercice.

L'avantage du format OEF est que les exercices sous ce format sont très faciles à créer. Mais il faut savoir que ces exercices, en particulier ceux créés par Createxo, ne sont pas aussi performants que des exercices propres à un système comme WIMS. Il est difficile d'imaginer, du moins pour l'instant, que les exercices WIMS tels que Q-Puzzle, la série Coïncidence, ou encore la série Tir, puissent être transformés en format oef.

Vous pouvez aussi consulter les exemples pour avoir une idée de la structure d'un fichier OEF.

Syntaxe de base

Le format d'un fichier OEF est très similaire à LaTeX. Il est composé d'instructions suivies d'un ou plusieurs paramètres. Une instruction est un mot précédé du caractère `\`, et les paramètres sont fermés dans des accolades. Par exemple, dans la phrase suivante

```
\answer{La réponse}{1; oui,non}{type=radio}
```

il y a une instruction ayant le nom `answer`, qui a 3 paramètres : `La réponse`, `1; oui,non` et `type=radio`. Dans

```
\choice{La réponse}{oui}{non}
```

il y a une instruction ayant le nom `choice`, qui a 3 paramètres : `La réponse`, `oui` et `non`.

Liste d'instructions

Voici une liste brève d'instructions reconnues par le gestionnaire OEF de WIMS. Voir aussi aide sur paramètres pour la syntaxe de définition de paramètres.

Instructions

instruction	nombre de paramètres	description	paramètres optionnels	mots d'option reconnus
`title`	1	définit le titre de l'exercice		
`language`	1	définit la langue de l'exercice, comme `en` ou `fr`		
`author`	1	définit l'auteur de l'exercice. Le mettre sous la forme `Prénom, Nom` (dans le cas de plusieurs auteurs, les séparer par des points-virgules).		
`email`	1	définit l'adresse électronique de l'auteur (dans le cas de plusieurs auteurs, les séparer par une virgule).		
`format`	1	format de l'énoncé		
`css`	1	définit le style css		
`keywords`	1	Mots clés de l'exercice (prendre de préférence les mots clés officiels séparés par des virgules)		
`credits`	1	permet d'inclure automatiquement un remerciement ou un crédit en fin d'exercice (les variables sont évaluées).		
`description`	1	description de l'exercice destinée à l'élève		
`observation`	1	description de l'exercice destinée à l'enseignant		
`precision`	1	précision en comparant la réponse de l'utilisateur avec la solution. Donnez un nombre positif n ici : la comparaison sera effectuée avec une tolérance de 1/n.		
`range`	1	zone de variables pour l'évaluation de fonction fournie par l'utilisateur. Doit être donnée sous forme `n1..n2`, où n1 est le point de départ, n2 le point d'arrivée.		
`computeanswer`	1	La commande `\computeanswer{ no }` précise que l'utilisateur doit lui-même faire les calculs et entrer la valeur finale. Si par contre, on met `\computeanswer{ yes }`, l'utilisateur peut entrer une formule comme `5*5`, laissant à l'ordinateur le soin de faire les calculs.		
`statement`	1	le paramètre est l'énoncé de l'exercice		
`answer`	2-5	définit une réponse libre. Le premier paramètre est le message pour la réponse, et le second est la bonne réponse. La réponse sera analysée selon des types (nombre, fonction, texte, etc).	`type` `option` `weight`	`reorder` `shuffle` `nonstop`
`choice`	3-5	définit un choix multiple. Le premier paramètre est le message pour le choix, le second les bons choix, et le troisième les mauvais choix. Les deux derniers paramètres peuvent (doivent) être une liste d'objets séparés par des virgules. Il est permis d'avoir plusieurs bons choix. Si un choix apparait à la fois comme bon et mauvais, il est pris pour bon.	`option` `weight`	`shuffle` `noidontknow`
`condition`	2-4	définit une condition spéciale pour l'évaluation de réponses libres. Le premier argument est un texte qui sera affiché lors de l'analyse de la réponse. Le second argument, on met la liste des conditions que la réponse de l'utilisateur doit satisfaire pour être considérée comme bonne.	`option` `weight`	`hide`
`solution`	1	donne une solution expliquée de l'exercice. Le gestionnaire OEF peut décider de montrer la solution ou pas à l'utilisateur, suivant le choix du niveau de difficulté pris par l'utilisateur. Ne peut apparaître qu'une seule fois dans un exercice.		

hint	1	donne une indication de l'exercice. Le gestionnaire OEF peut décider de montrer l'indication ou non, suivant le niveau de difficulté. Ne peut apparaître qu'une seule fois dans un exercice.		
help	1	donne une aide à l'exercice. Cette aide sera toujours accessible à l'utilisateur, dans une fenêtre `popup'. Ne peut apparaître qu'une seule fois dans un exercice.		
feedback	2	affiche un commentaire quand la réponse tombe sous une certaine condition. Peut normalement être utilisé pour avertir d'une erreur typique.		
steps	1	sert à définir les questions apparaissant à chaque étape ; doit être mis avant la commande `statement` (voir la variable `\step`). Ne peut apparaître qu'une seule fois dans un exercice.		
nextstep	1	sert à définir de manière dynamique les questions qui devront être posées ; doit être mis avant la commande `statement` (voir la variable `\step`). Can be used only one time in the exercise.		
conditions	1	permet d'indiquer les numéros des conditions utiles pour l'exercice servant à contrôler les réponses de l'utilisateur.		
latex	1	permet d'écrire une version en latex de l'exercice utilisant les variables définies dans l'exercice et pouvant être téléchargées dans la version imprimable de l'exercice (accessible uniquement par les développeurs ou les enseignants d'une classe). Il est conseillé de mettre l'énoncé dans un environnement latex prédéfini `\begin{statement} \end{statement}` et la solution dans l'environnement `\begin{sol} ... \end{sol}`.		

Exemples d'exercices interactifs sous le format OEF

Voici quelques exemples d'exercices interactifs qu'on peut créer par Createxo.

1. **Longueur de vecteur 2D**, un simple calcul de longueur d'un vecteur dans le plan. Voici le source complet de cet exercice.

```
\title{Norme de vecteur 2D}
\language{fr}
\computeanswer{no}
\format{html}

\integer{x=random(-10..10)}
\integer{y=random(-10..10)}
\real{norm=sqrt((\x)^2+(\y)^2)}
\statement{Quelle est la longueur du vecteur (\x,\y) dans R<sup>2</sup>?}

\hint{La longueur d'un vecteur (x,y) est égale à sqrt(x^2+y^2).}
\answer{La longueur}{\norm}

```

Éditeur actif

Dans cet exercice, on a défini 2 entiers aléatoires, x et y, qui sont les coordonnées du vecteur. Ensuite un troisième paramètre, cette fois réel, est défini par la formule de longueur. L'exercice prend une réponse libre sous le nom de ``La longueur'', et la bonne réponse doit être la valeur du troisième paramètre ``norm''. Une indication est préparée dans l'exercice, qui rappelle la formule de longueur.

Vous pouvez charger cet exemple dans le menu pour le tester. (Vous pouvez aussi copier la source dans le menu en mode brut.)

2. **Trace de matrice 2x2**, calcul de trace d'une matrice. La matrice est formatée par TeX, pour une meilleure présentation. Voici le source complet de l'exercice.

```
\title{Trace de matrice 2x2}
\language{fr}
\computeanswer{yes}
\format{html}

\integer{lim=20}
\integer{a=random(-\lim..\lim)}
\integer{b=random(-\lim..\lim)}
\integer{c=random(-\lim..\lim)}
\integer{d=random(-\lim..\lim)}
\integer{trace=(\a)+(\d)}
\statement{Calculer la trace de la matrice
\([\a,\b;\c,\d]\).}

\answer{La trace}{\trace}

```

Éditeur actif

On a d'abord défini un entier `lim` qui permettra d'encadrer les valeurs aléatoires a, b, c, d qui sont les éléments de la matrice. La trace est définie par la somme des éléments sur la diagonale. Faites attention à la définition `trace=(\a)+(\d)` : les paires de parenthèses sont nécessaires, car la substitution est littérale. Si on définit `trace=\a+\d` et si a et d prennent les valeurs de 3 et -15 respectivement, on aurait `trace=3+-15`, ce qui est une mauvaise expression mathématique.

Remarquons que dans cet exercice, les réponses non calculées sont admises (telles 2+15 ou 3*105).

Vous pouvez charger cet exemple dans le menu pour le tester. (Vous pouvez aussi copier la source dans le menu en mode brut.)

Les formats de l'énoncé

Vous pouvez écrire l'énoncé de l'exercice sous l'un des deux formats: html ou TeX. A cause du développement des capacités du format html (voir ci-

dessous), le format TeX n'est plus indispensable et est à éviter désormais.

Si votre exercice ne contient pas de formules mathématiques compliquées (par exemple des matrices), il est vivement conseillé d'utiliser le format html. L'exécution sera plus rapide, et il y aura moins de problème de compatibilité, surtout dans le futur.

Vous pouvez utiliser le format html, si l'énoncé de l'exercice ne contient rien d'autre que du simple texte. Il faut surtout prendre soin de remplacer toute apparence du signe `<` par la chaîne `<` , pour éviter qu'il soit interprété comme l'ouverture d'un tag html. En dehors de cela, les tags html peuvent être utilisés, en particulier ceux pour les indices (`<sub>` et `</sub>`) et les exposants (`<sup>` et `</sup>`).

En particulier, les liens http peuvent être insérés dans l'énoncé. Par exemple, on peut inclure des dessins dans l'exercice, sans pour autant avoir besoin de soumettre ces dessins au serveur. (Il vous suffit de déclarer la source du dessin par une adresse web complète.)

Vous pouvez aussi inclure des symboles et formules mathématiques très facilement dans votre exercice même s'il est en format html. Si cela ne vous satisfait encore pas, le format TeX est disponible. Pour l'utiliser, vous devez connaître la syntaxe de TeX.

En format html, vous pouvez aussi inclure des champs de réponse et des dessins dans votre énoncé.

A remarquer aussi que les autres champs de l'exercice (prompts de réponses, indication et solution) acceptent uniquement le format html. Les symboles et formules mathématiques peuvent pourtant être insérés dans l'indication et la solution.

Paramètres aléatoires dans un exercice interactif

L'utilisation de paramètres aléatoires rendra votre exercice beaucoup plus intéressant, car ce sera un exercice différent chaque fois qu'il est redemandé.

Par exemple, la ligne suivante définit un paramètre sous le nom de `x1` , dont la valeur sera un entier aléatoire entre -10 et 10 (inclusif) :

```
\integer{x1=random(-10..10)}
```

Ce paramètre aléatoire peut ensuite être invoqué par le mot `\x1` , dans l'énoncé, les réponses, l'indication et la solution. C'est-à-dire, chaque mot `\x1` dans ces textes sera remplacé par la valeur aléatoire du paramètre. Cette substitution prend aussi effet dans les définitions d'autres paramètres qui suivent celle de `x1` .

Supposons maintenant que vous avez entré

```
\integer{x1=random(-10..10)}
\integer{y1=\x1+3}
```

dans le champ de la définition de paramètre, et la question

```
Calculez la multiplication de \x1 par \y1.
```

dans l'énoncé de l'exercice. Supposons que sur une demande de l'exercice, une valeur aléatoire `-7` est attribuée à \x1. Alors le paramètre suivant \y1 prendra la valeur -4, et l'énoncé de l'exercice sera présenté sous la forme

Calculez la multiplication de -7 par -4.

Vous pouvez ensuite définir une réponse numérique au nom de `Le produit` , ayant pour bonne solution `(\x1)*(\y1)` . (Remarquez qu'ici les parenthèses sont nécessaires car la substitution se fera de façon littérale.) >

Quelques autres exemples de paramètres [liste complète]

Définition	Effet
`\real{x=random(-5..5)}`	\x sera un nombre réel aléatoire entre -5 et 5
`\real{a=random(-5,-3,0.3,4)}`	\a sera un nombre réel pris aléatoirement parmi -5,-3,0.3 et 4
`\complex{z=(1+2*i)^3}`	\z sera le nombre complexe z=(1+2*i)^3
`\text{sign=random(+,-)}`	\sign sera un signe aléatoire: + ou -
`\integer{n=3*exp(\a)}`	\n sera l'entier le plus proche de $3*e^{\a}$ (il dépend de la valeur de \a)
`\function{f=random (x^2+1,sin(x),log(x))}`	\f sera une fonction aléatoire: soit x^2+1, soit sin(x), soit log(x)
`\real{a=evalue(x^2+sin(y),x=3,y=4)}`	Evaluation de la fonction x^2+sin(y), pour x=3, y=4
`\real{r=solve(x^3-3*x+1,x=0..1)}`	\r sera la racine simple de x^3-3x+1 entre 0 et 1
`\function{h=simplify(x^5*y^3*x^2/y)}`	Expression simplifiée : x^7y^2
`\function{g=diff(sin(x)+cos(y),x)}`	\g sera la dérivée de sin(x)+cos(y) par rapport à x
`\function{F=int(x^2+3*x+1,x)}`	\F sera une primitive de x^2+3*x+1, le terme constant n'étant pas garanti d'être toujours le même
`\real{a=int(t^2+3*t+1,t=0..1)}`	\a sera l'intégrale numérique de x^2+3*x+1, de 0 à 1
`\text{f=htmlmath(2*x^2+3*x)}`	\f sera rendu en html comme: $2x^2+3x$
`\text{f=texmath(2*x^2+3*x)}`	\f sera le source TeX de l'expression.
`\integer{n=items(a,b,c,d,e,f)}`	\n sera le nombre d'articles (ici c'est 6) dans la liste {a,b,c,d,e,f}
`\text{i=item(3,a,b,c,d,e,f)}`	\i sera l'article numéro 3 de la liste {a,b,c,d,e,f} (donc c).
`\text{s=shuffle(6)}`	\s sera la liste des 6 entiers 1,2,...,6, dans un ordre aléatoire.
`\text{s=shuffle(a,b,c,d,e)}`	\s sera les lettres {a,b,c,d,e} dans un ordre aléatoire.

`\matrix{m=1,2,3`
`4,5,6`
`7,8,9}`
	\m sera la matrice de 3 lignes et 3 colonnes.

`\text{t=asis(Comment ça va ? matrix(1,2,3))}`	\t est la chaîne comme elle est écrite, sans transformation ni conditionalité.

Paramètres conditionnels : vous pouvez écrire

```
\text{ttt=_condition?_def1}
\text{ttt=_condition?_def1:_def2}
```

Dans ce cas, `ttt` aura la valeur `_def1` si `_condition` s'avère vraie, ou `_def2` sinon (dans la seconde syntaxe). Liste de conditions

La position relative d'une définition par rapport à l'énoncé est importante : si une variable est définie APRÈS l'énoncé, l'évaluation de la variable aura lieu uniquement APRÈS que l'utilisateur ait répondu à la question. Dans ce cas, la définition peut utiliser les réponses données par l'utilisateur, via `\reply1`, `\reply2`, etc. Et la variable ainsi définie peut être utilisée dans les conditions de test ou les feedbacks.

Liste de types de variables

Type	Exemple	Signification
`real`	`\real{x=random(-5..5)}`	\x est un nombre réel aléatoire entre -5 et 5
`complex`	`\complex{z=(1+2*i)^3}`	\z est le nombre complexe z=(1+2*i)^3
`text`	`\text{a=1 2 3}`	\a est le texte donné sans aucune transformation
`integer`	`\integer{n=3*exp(\a)}`	\n est l'entier le plus proche de $3*e^{\a}$ (il dépend de la valeur de \a)
`rational`	`\rational{x=2*5/6}`	\x est le nombre rationnel (exact) 5/3
`function`	`\function{f=2*x^2}`	la fonction 2 x^2

`matrix`	`\matrix{m=1,2,3`
	`4,5,6`
	`7,8,9}`
		\m est la matrice de 3 lignes et 3 colonnes.
		\m[2..-1;] revoie les lignes de \m de 2 à la dernière ;
		\m[;2,3] renvoie les colonnes 2 et 3 de \m.

Variables prédéfinies

Variable	Exemple	Signification
`reply`	`reply1 reply2 ...`	variable contenant la réponse de l'élève à la question de type `reply` numéro 1
`choice`	`choice1 choice2 ...`	variable contenant la réponse de l'élève à la question de type `choice` numéro 1
`step`	`step`	variable donnant le numéro de l'étape (dans le cas où une commande de type
`sc_reply`	`sc_reply1 sc_reply2` `...`	variable entre 0 et 1 indiquant si la réponse à la question est juste ou non
`reply_`	`reply_1 reply_2 ...`	variable contenant la réponse de l'élève formatée comme elle apparait dans l'analyse de la réponse
`help_subject`	`help_subject`	variable contenant l'identificateur de l'aide introduite par la méthode spéciale `help`.
`oef_firstname`	`oef_firstname`	Prénom du participant, par défaut `Visiteur Inconnu`.
`oef_lastname`	`oef_lastname`	Nom du participant, par défaut `Anonyme`.
`oef_login`	`oef_login`	Identifiant du participant, par défaut `anonymous`.
`oef_now`	`oef_now`	Temps sous la forme aaaammjj:hh:mm:ss (20110515.22:19:25)
`oef_lang`	`oef_lang`	Langage du participant ou visiteur (par exemple, `en`, `fr`, `es`)

Fonctions aléatoires

Fonction	Exemple	Signification
`random( .. )`	`random(-5..5)`	un nombre au hasard entre -5 et 5.
`randint( .. )`	`randint(-5..5)`	un entier au hasard entre -5 et 5 (bornes comprises).
`random( )`	`random(1,2,3,a,b,c)`	un item au hasard parmi {1,2,3,a,b,c}
`shuffle( )`	`shuffle(6)`	une liste de 6 entiers 1,2,...,6, dans un ordre aléatoire.
`shuffle( , )`	`shuffle(a,b,c,d,e)`	la liste des lettres {a,b,c,d,e}, dans un ordre aléatoire. Attention, si le premier mot est `even` ou `odd`, la permutation effectuée sera respectivement paire ou impaire (aussi, cela ne peut être le premier mot de la liste à permuter).
`randomitem( )`	`randomitem(\list)`	un item au hasard de la liste \list (items séparés par des virgules).
`randitem( )`	`randitem(+,-)`	un signe aléatoire: + ou -; `randitem(\list)` est un item au hasard de la liste `\list` (items séparés par des virgules).
`randomrow( )`	`randomrow(\mat)`	une ligne au hasard de la matrice \mat.

Données et manipulations de listes

Fonction	Exemple	Signification

items( )	items(a,b,c,d,e,f)	le nombre d'items de la liste {a,b,c,d,e,f} (6 dans cet exemple)
item( )	item(3,a,b,c,d,e,f)	item numéro 3 de la liste {a,b,c,d,e,f} (ici c) ; item(3,\ll) est l'item numéro 3 de la liste \ll (de manière équivalente : :\ll[3])
item(.. ,)	item(2..5,a,b,c,d,e,f)	items numéros 2 à 5 de la liste {a,b,c,d,e,f} (ici b,c,d,e)
item([,],)	item([2,4],\ll)	items numéros 2 et 4 de la liste \ll (de manière équivalente \ll[2,4])
position( )	position(make,do,go,make,take)	numéros des positions de l'item `make' dans la liste {do,go,make,take} (ici 3)
rows( )	rows(\m)	nombre de lignes de la matrice \m
row(,)	row(2,\m)	ligne numéro 2 de la matrice \m (de manière équivalente : \m[2;])
row(.. ,)	row(2..5,\m)	la matrice extraite de \m formée des lignes numéros 2 à 5 (de manière équivalente :
row([,],)	row([1,3],1,2,3 3,4,5 5,6,7)	la matrice extraite de la matrice 3×3 formée de la première ligne et de la troisième ligne
row(,)	row(column 1 > 1 and column 2 = good,\mat)	la matrice extraite de \mat formée des lignes dont la colonne 1 est > 1 et dont la colonne 2 est le mot `good'
randomitem( )	randomitem(\list)	un item au hasard de la liste \list (items séparés par des virgules).
randomrow( )	randomrow(\mat)	une ligne au hasard de la matrice \mat.
column(,)	column(2,\m)	les items de la colonne numéro 2 de la matrice \m, le résultat est une liste séparée par des virgules (de manière équivalente :\m[;2])
column(.. ,)	column(2..5,\m)	la matrice extraite de \m formée des colonnes numéros 2 à 5 (de manière équivalente :
column([,],)	column([1,3],1,2,3 3,4,5 5,6,7)	la matrice extraite de la matrice 3×3 formée de la première et de la troisième colonne
asis( )	asis(Comment ça va? item(1,2,3))	la chaîne de caractères telle qu'elle est sans aucune substitution ou interprétation.
htmlmath( )	htmlmath(2*x^2+3*x)	la manière la meilleure possible de rendre l'expression en html: $2x^2+3x$
texmath( )	texmath(2*x^2+3*x)	le source TeX de l'expression

Fonctions mathématiques

Fonction	Exemple	Signification
evalue( )	evalue(x^2+sin(y),x=3,y=4)	évaluation de la fonction x^2+sin(y) en x=3, y=4
solve( )	solve(x^3-3*x+1,x=0..1)	la racine simple de x^3-3x+1 entre 0 et 1
simplify( )	simplify(x^5*y^3*x^2/y)	expression simplifiée : x^7y^2
diff( )	diff(sin(x)+cos(y),x)	la dérivée de sin(x)+cos(y) par rapport à x
int(,)	int(x^2+3*x+1,x)	primitive de x^2+3*x+1, le terme constant étant indéterminé
int(, = ..)	int(t^2+3*t+1,t=0..1)	l'intégrale numérique de x^2+3*x+1, entre 0 et 1
det( )	det(\mat)	le déterminant de la matrice \mat
abs( )	abs(-3.4)	valeur absolue (equivalent : fabs( ))
sqrt( )	\real{a=sqrt(32)}	racine carrée
binomial(,)	\integer{a=binomial(9,3)}	le coefficient binomial (pour des coefficients inférieurs à 10^7 sinon utiliser la fonction de pari \text{a=pari(binomial(50,10))}
ceil( )	\real{a=ceil(3.4)}	le plus petit entier supérieur
floor( )	\real{a=floor(3.4)}	le plus grand entier inférieur
rint( )	\real{a=rint(3.4)}	l'entier le plus proche (equivalent : round( ))
e	\real{a=e^2}	constante mathématique *e* ; (equivalent : E)
erf( )	\real{a=erf(3.4)}	Fonction erf
erfc( )	\real{a=erfc(3.4)}	Fonction erfc
Euler	\real{a=Euler}	constante d'Euler : EULER euler
exp( )	\real{a=exp(4)}	exponentielle
factorial( )	\integer{a=factorial(4)}	factorielle
Inf	\real{a=Inf + 3}	l'infini
gcd(,)	\integer{a=gcd(4,6)}	pgcd de deux entiers
lcm(,)	\integer{a=lcm(4,6)}	ppcm de deux entiers
%	\integer{a=5%2}	reste de la division euclidienne
max(,)	\real{a=max(4,6)}	maximum de deux nombres
min(,)	\real{a=gcd(4,6)}	minimum de deux nombres
lg( )	\real{a=log10(10^4)}	log en base 10 (equivalent : log10)
lgamma( )	\real{a=lgamma(e^(24))}	log de la fonction Gamma

ln( )	\real{a=ln(e^4)}	log népérien (equivalent : log)
log2( )	\real{a=log2(2^4)}	log en base 2
pow( )	\real{a=pow(3,0.6)}	puissance, équivalent à 3^0.6
sgn( )	\integer{a=sign(-4)}	signe de la valeur (equivalent : sign)
PI	\real{a=sin(Pi)}	constante mathématique π, (equivalent : Pi, pi)
sin	sin(3)	fonctions trigonométriques (autres fonctions : tg tan sec (1/sin) cot cotan cotan ctg csc (1/cos))
acos	acos(0.5)	fonctions trigonométriques réciproques (autres fonctions : acos arccos acos arcsin asin arctan atan arctg atan)
sh	sh(4)	fonctions hyperboliques (autres fonctions : sh sinh tanh tanh th ch cosh coth cotanh)
Argch	Argch(4)	fonctions hyperboliques réciproques (autres fonctions : Argch acosh argch Argsh asinh argsh Argth atanh argth)

Possibilités avancées

Fonction	Exemple	Signification
pari	pari(factor(2^101-1))	appel de PARI/GP: ici pour factoriser un entier (utiliser en général de préférence avec \text{ })
maxima	maxima(integrate(x^2+1,x);)	appel de Maxima: ici pour intégrer une fonction
yacas	yacas(Taylor(x,0,10) cos(x^2+x+1))	appel de Yacas: ici pour calculer un développement de Taylor
wims	wims(sort items \list) wims(listintersect \list1 and \list2)	utilisation de commandes wims (deux exemples : la première utilise la commande wims !sort pour ordonner les items de la liste \list, la seconde utilise la commande wims !listintersect pour obtenir les items communs des listes \list1 et \list2)
draw	draw(pixel_size_x,pixel_size_y draw_source)	dessiner, le source est le même que pour la commande \draw, la première ligne étant formée de la taille de l'image en pixels. La sortie est l'adresse URL de l'image.
slib	slib(matrix/invertible 3,5)	bibliothèque de scripts, par exemple, ici une matrice inversible 3x3 dont les coefficients sont inférieurs à 5.
teximg	teximg(\(\displaystyle{\frac{3}{4}}\))	crée une image d'un texte mathématique. La sortie est l'adresse URL de l'image. Elle peut être recopiée dans un dessin créé avec Flydraw.

Réponses à un exercice

Un exercice OEF peut accepter les réponses des utilisateurs de manière très variée. Outre le choix multiple ou la réponse libre, on trouve de nombreuses autres possibilités. La réponse donnée par l'utilisateur est analysée selon la nature du type. D'autre part, il est possible d'afficher des commentaires ciblés quand la réponse tombe sous une certaine condition (pour avertir d'une erreur typique par exemple). Actuellement, jusqu'à 100 réponses simultanées peuvent être demandées dans un exercice sous l'implémentation actuelle.

La syntaxe est la suivante (les 3 derniers champs sont optionnels) :

```
\answer{Texte}{\reponse}{type=...}{option=...}{weight=...}
```

Le paramètre `type`

Chaque réponse peut prendre indépendamment l'un des types suivants.

Types de base

- **Auto.** (nom: `default`)
  Le logiciel peut déterminer le type de la réponse suivant la bonne réponse donnée. Cette capacité de détermination automatique est assez limitée et peut distinguer seulement fonctions numériques/nombres/équations/textes, donc si possible, il est préférable de préciser le type.
- **Texte brut.** (nom: `raw`)
  Un texte à comparer avec la bonne réponse donnée. Les options suivantes permettent de contrôler comment le texte doit être transformé avant d'être comparé. Si aucune option n'est indiquée, seuls les espaces en début et en fin du texte sont enlevés avant comparaison. Il faut donc faire très attention en utilisant ce type de réponses : par défaut, la réponse est considérée comme fausse même s'il y a un espace de différence. Ce type est destiné aux auteurs désirant analyser eux-même la réponse.

Options

Syntaxe	Signification
noaccent	enlève les accents sur les lettres.
nocase	transforme les lettres en minuscule.
nodigit	remplace les chiffres par des espaces.
nomathop	remplace les opérateurs mathématiques par des espaces
noparenthesis	remplace les parenthèses par des espaces.
nopunct	remplace les ponctuations par des espaces.
noquote	remplace les apostrophes ou guillemets (simple et double) par des espaces.
nospace	enlève tous les caractères d'espace (y compris ceux provenant de remplacement d'autres caractères).
reaccent	permet les lettres accentuées précédées de \

singlespace	traite toutes les chaînes de caractères d'espaces comme un seul espace.

Remarque. Ce type de réponse accepte l'option symtext. Si le mot `symtext` est déclaré dans l'option de la réponse, l'analyse de la réponse sera exactement comme pour symtext, en particulier sans aucun traitement préalable des textes. Et toutes les options symtext seront comprises dans ce cas.

- **Nombre.** (nom: `numeric`)
Comparaison de nombres réels à la précision définie par l'auteur. Le type `numeric` compare la réponse donnée par l'élève à la bonne réponse de l'enseignant avec une précision définie.
La syntaxe générale de ce type de réponse est :

```
\answer{un commentaire}{0.42}{type=numeric}
\answer{un commentaire}{\rep}{type=numeric}{option=comma}
```

où `\rep` est la bonne réponse définie auparavant ou une variable n'ayant pas été définie dans le cas où l'on désire analyser soi-même la réponse. Si `\rep` n'a pas été définie, vous devez faire l'analyse vous-même en utilisant les conditions (`\condition`). Ce qui suit ne concerne que le cas d'une variable déjà définie et ayant une valeur.
La valeur numérique attendue est estimée par défaut avec une erreur relative :

$$|\text{reponse} - \text{bonne\_reponse}| \leq M(\text{prec})$$

avec

$$M(\text{prec}) = \frac{\max(|\text{reponse}| + |\text{bonne\_reponse}|, \frac{1}{\text{prec}})}{\text{prec}}$$

Dans la formule précédente, `prec` est donnée dans le champ `\precision` , par défaut sa valeur est de 1000. Autrement dit, si la réponse est proche de la bonne réponse, le quotient de la différence par la bonne réponse (erreur relative) doit être inférieure à 2/prec.
Pour qu'une réponse soit considérée comme **bonne avec une mauvaise précision**, il faut remplacer `prec` par $\sqrt{\text{prec}}$.
Les options possibles sont les suivantes :
  - `comma` : l'écriture des nombres décimaux avec une virgule décimale est acceptée et la bonne réponse est donnée avec le même "séparateur" décimal que celui donné par l'élève.
  - `absolute` : la précision est absolue et le calcul d'erreur s'effectue par la relation :

$$|\text{reponse} - \text{bonne\_reponse}| < \frac{1}{\text{prec}}.$$

Pour que la réponse soit considérée comme **bonne avec une mauvaise précision**, il faut remplacer $\frac{1}{\text{prec}}$ par $\frac{10}{\text{prec}}$.

Un exemple de calcul est proposé pour illustrer les précisions absolues et relatives. La valeur numérique exacte attendue est 0.42 et la valeur `prec` du champ `\precision` est égale à 1000.

Réponse	Précision relative	Précision absolue	Différence avec 0.42	M(prec)	$M(\sqrt{\text{prec}})$	$\frac{1}{\text{prec}}$	$\frac{10}{\text{prec}}$
0.4205	♦	♦	0.0005	0.0008405	0.02658	0.001	0.01
0.4192	♦	♦	0.0008	0.0008392	0.02654	0.001	0.01
0.4191	♦	♦	0.0009	0.0008391	0.02653	0.001	0.01
0.4190	♦	♦	0.001	0.000839	0.02653	0.001	0.01
0.4	♦	♦	0.02	0.00082	0.02593	0.001	0.01
0.39	♦	♦	0.03	0.00081	0.02561	0.001	0.01
500000	♦	♦	499999.58	500.00042	15811.402	0.001	0.01

Réponse	Précision relative	Précision absolue	Différence avec 505.42	M(prec)	$M(\sqrt{\text{prec}})$	$\frac{1}{\text{prec}}$	$\frac{10}{\text{prec}}$
505.4205	♦	♦	0.0005	1.0108405	31.96558	0.001	0.01
505.4192	♦	♦	0.0008	1.0108392	31.96554	0.001	0.01
505.4191	♦	♦	0.0009	1.0108391	31.96554	0.001	0.01
505.4190	♦	♦	0.001	1.010839	31.96554	0.001	0.01
505.4	♦	♦	0.02	1.01082	31.96494	0.001	0.01
505.39	♦	♦	0.03	1.01081	31.96462	0.001	0.01

- **Fonction numérique.** (nom: `function`) La réponse est évaluée en tant que fonction et la comparaison est effectuée pour des valeurs de variable dans une zone définie par l'auteur, à la précision définie par l'auteur.

  La bonne réponse est donnée par la fonction. Si l'élève donne une réponse contenant des variables qui ne sont pas dans la fonction, il lui est demandé de resoumettre sa réponse. Pour éviter cela, il est possible d'ajouter une suite de variables qui seront admises. Cela est nécessaire en particulier lorsque la fonction est aléatoire et que son nombre de variables peut varier.

```
\answer{}{x^2,x,y}{type=function}
```

La zone de valeurs des variables est définie dans le champ \range{} qui est par défaut l'intervalle [-5,5]. Cependant, il est possible de préciser pour chaque variable un intervalle servant à la comparaison comme dans les exemples ci-dessous.

```
\answer{}{log(x)+y,x=[1,5],y}{type=function}
```

Dans ce cas, la réponse est testée pour 10 valeurs aléatoires de x entre 1 et 5 et 10 valeurs aléatoires de y entre -5 et 5.

```
\answer{}{log(x)+log(-y),x=[1,5],y=[-5,-1]}{type=function}
```

Dans ce cas, la réponse est testée pour 10 valeurs aléatoires de x entre 1 et 5 et 10 valeurs aléatoires de y entre -5 et -1.

En particulier, dans le cas de plusieurs questions dans un même exercice, l'intervalle peut être différent selon la question.

```
\answer{}{log(x),x=[1,5]}{type=function}
\answer{}{log(-x),x=[-5,-1]}{type=function}
```

L'option possible est `integer`.

```
\answer{}{1/(x+9)+1/(x-9)+1/x}{type=function}{option=integer}
```

Dans ce cas, les valeurs de test sont entières dans l'intervalle donnée. Il est aussi possible de fixer les valeurs de test en donnant explicitement la liste :

```
\answer{}{1/(x-3)+1/(x-2)+1/x,x=[-8,-7,-6,-4,4,5,6,7,8,9]}{type=function}{option=integer}
```

Cela est à manier avec prudence, en particulier, il est recommandé de donner au moins 10 valeurs.

- **Equation formelle.** (nom: `equation`)
  La réponse va être évaluée en tant qu'équation et la comparaison est effectuée pour des valeurs de variable dans une zone définie par l'auteur, jusqu'à la précision définie par l'auteur. La réponse `a = 0` est traitée comme la réponse `a`. L'option `eqsign=yes` oblige à écrire `a = 0`.
- **Expression mathématique.** (noms : `algexp`, `litexp` et `formal`)

  Une expression mathématique est comparée à la bonne réponse donnée selon différents critères d'identification. On peut mettre plusieurs bonnes réponses en les séparant par des virgules.

  Pour `algexp` (expression algébrique), il y a des identifications limitées pour la comparaison. Par exemple, (x+1)(x-1) n'est pas accepté quand la bonne réponse est x^2-1, sin(x)^2+cos(x)^2 non plus quand la bonne réponse est 1. Par contre, x-y*y et -y^2+x sont considérés comme les mêmes. Il est utile pour forcer les étudiants à faire les manipulations d'expressions eux-mêmes.

  Pour `litexp` (expression littérale), la comparaison est littérale sans aucune simplification algébrique. Par exemple, x+y n'est pas identifié à y+x, ni 3/2 avec 6/4. Mais 2x et 2*x sont identifiés et les espaces sont enlevés avant comparaison. A utiliser avec beaucoup de précaution.

  Le type `formal` (expression formelle) permet des comparaisons numériques exactes. Tout ce qui est égal exactement à la bonne réponse est accepté, mais l'approximation n'est pas admise.

- **Texte (sensible ou non à la casse : majuscules/minuscules et approximatif).** (noms : `case`, `nocase` et `atext`)
  Les types `case`, `nocase` et `atext` comparent la réponse donnée par l'élève aux bonnes réponses textes données par l'enseignant. La syntaxe générale de ce type de réponse est :

```
\answer{}{BlAnC|WhItE}{type=case}{option=noreaccent}
\answer{}{BlAnC}{type=nocase}
\answer{Commentaire}{BonneRéponse|Synonyme1|Synonyme2;MauvaiseRéponse1|MauvaiseRéponse2}{type=atex
\answer{}{BlAnC|WhItE;rouge|bleue}{type=atext}
```

  Vous pouvez définir des synonymes dans la bonne réponse. Pour cela, ajoutez simplement les synonymes après la bonne réponse (standard), précédés de la barre verticale (tube) `|`. On peut définir un nombre quelconque de synonymes.

  Voici une description de chacun des types et de leurs différences.
  - `case` : Chaque lettre de la réponse doit être identique à l'une des bonnes réponses attendues. Les expressions du type e' sont transformées en é (réaccentuation) pour les élèves qui utilisent un clavier sans accent. Pour l'éviter, utiliser l'option `noreaccent`.
  - `nocase` : La comparaison ne tient pas compte des différences entre lettres majuscules et minuscules. Les accents ne sont pas évalués.
  - `atext` : Dans le cas du type atext, la comparaison n'est faite que sur les éléments essentiels des textes. Sont ignorés
    - les différences majuscule/minuscule,
    - certaines différences singulier/pluriel (s en fin de mot),
    - les accents sur les lettres,
    - les mots très communs (de, le, un, ...).

  Pour que la réponse de l'élève soit obligatoirement contenue dans un champ lexical donné, on peut définir des mauvaises réponses en utilisant le point virgule `;` comme séparateur puis le tube `|` pour mettre plusieurs mauvaises réponses. Si la réponse donnée par l'élève n'est pas incluse dans ce champ lexical, il est demandé à l'élève de compléter à nouveau la réponse pour forcer l'utilisation d'un des mots du champ lexical. Les signes de ponctuation seront ignorés dans cette liste et les mots sont comparés avec le même niveau de tolérance que pour les bonnes réponses.

  Remarque. Ce type de réponse accepte l'option symtext. Si le mot `symtext` est déclaré dans l'option de la réponse, l'analyse de la réponse sera exactement comme pour symtext, en particulier sans aucun traitement préalable des textes. Et toutes les options symtext seront comprises dans ce cas.

- **Choix multiples contrôlables par l'auteur**: (noms : `checkbox`, `click`, `menu`, `radio`, `mark`, `flashcard`, `multipleclick`)
  Ces types de réponses sont utilisés pour remplacer l'entrée standard de choix multiple (`\choice`), vous donnant plus de contrôle sur l'apparence. La syntaxe générale est la suivante (avec Type l'un des types précédents :

```
\answer{Commentaire}{numéros_bons_choix;Liste_des_différents_choix}{type=Type}{option=}
```

  Dans le cas d'utilisation de la commande `\embed`, il est possible de placer précisément le k-ième item en écrivant `\embed{r n ,k}` avec `n` le numéro de la question (sauf pour `multipleclick`). Par exemple,

```
\for{j=1 to \N}{ \embed{r1,\j}}
```

  où \N est le nombre d'items. Tous les items proposés sont alors affichés lors de la correction. Dans le cas contraire, seule la réponse de l'élève est affichée en vert ou en rouge selon qu'elle est juste ou fausse.

  Pour disposer les éléments de réponses dans une liste verticale, un code du type suivant peut être utilisé (à placer dans le \statement) :

```
<ul>\for{h=1 to 6}{<li>\embed{reply4,\h}</li>}</ul>
```

  L'affichage de la correction suit les règles suivantes (lorsque cela est pertinent):
  - les bonnes réponses sont en vert,
  - les mauvaises sont en rouge,
  - les réponses oubliées sont en bleu

  L'option `nolegend` permet de désactiver la légende des couleurs). Description de chacun des types.
  - **Cocher une ou des cases** (noms : `radio` et `checkbox`)
    Le type `radio` (resp. `checkbox`) analyse la case cochée (resp. les cases cochées) par l'élève et compare à la bonne réponse. Par défaut la note est 0 dès que sa réponse ne s'accorde pas exactement à la réponse attendue dans le cas `radio` et si sa réponse n'est pas une des bonnes réponses dans le cas `checkbox` (sauf avec l'option `split`).
    La syntaxe générale est :

```
\answer{Commentaire}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=radio}
\answer{question 2}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=radio}{option=shuffle}
\answer{question 1}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=checkbox}
\answer{question 3}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=checkbox}{opt
\answer{question 4}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=checkbox}{opt
```

    - `radio` : L'élève ne peut sélectionner qu'une réponse et les options possibles sont uniquement `shuffle` ou `sort`.
    - `checkbox` : L'élève peut sélectionner une ou plusieurs réponses. Les options possibles sont plus variées :
      - `shuffle` : Ordre des réponses aléatoires
      - `sort` : Réponses dans l'ordre alphabétique
      - `split` : Note partielle à la question. Par défaut 1 mauvais choix est compensé par 2 bons. Une telle politique est nécessaire, entre autres, pour dissuader de prendre tous les choix disponibles dans le cas où beaucoup d'entre eux sont bons.
      - `eqweight` : Score partiel équivalent. Un bon choix est compensé par un mauvais. Cette politique en apparence juste peut conduire à des effets pervers !
  - **Sélectionner dans un menu** (nom : `menu`)
    La syntaxe générale de ce type de réponse est :

```
\answer{Commentaire}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=menu}
\answer{question 2}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=menu}{option=shuffle}
\answer{question 3}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=menu}{option=
\answer{question 4}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=menu}{option=
\answer{question 2}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=menu}{option=shuffle n
```

    - `menu` sans l'option `mutiple` : L'élève ne peut sélectionner qu'une réponse et les options possibles sont uniquement `shuffle` ou `sort`.
    - `menu` avec l'option `multiple` : L'élève peut sélectionner plusieurs réponses en maintenant la touche ctrl (dépend éventuellement du navigateur). Les options possibles sont plus variées :
      - `shuffle` : Ordre des réponses aléatoires
      - `sort` : Réponses dans l'ordre alphabétique
      - `multiple` : Hauteur de la fenêtre

```
\answer{question 2}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=menu}{option=
```

        L'option ci-dessus affiche 3 choix parmi les 7 mais un ascenseur permet de sélectionner les autres possibilités.
    - `split` : Note partielle à la question. Par défaut 1 mauvais choix est compensé par 2 bons. Une telle politique est nécessaire, entre autres, pour dissuader de prendre tous les choix disponibles dans le cas où beaucoup d'entre eux sont bons.
    - `eqweight` : Score partiel équivalent. Un bon choix est compensé par un mauvais. Cette politique en apparence juste peut conduire à des effets pervers !
  - **Surligner au marqueur** (nom : `mark`)
    Le type `mark` compare les mots ou expressions cliqués par l'utilisateur avec une liste de bonnes réponses définies par l'enseignant. Il est principalement destiné à une utilisation dans le texte de l'énoncé. Le ou les mots changent de couleur lorsque l'utilisateur clique dessus, pour indiquer ses choix. Par défaut l'analyse est réalisée de manière stricte. Pour écrire les énoncés, il faut séparer chaque terme à analyser par une virgule. Par conséquent, remplacer les virgules du texte par le code html , ou par \(,). En procédant ainsi la virgule n'interfére pas avec l'analyse de la réponse.

```
\text{enonce=Pour, commercialiser,et, donc, vendre, ses, produits\(,), il, a, besoin, d'un, vé
```

    Dans l'exemple précédent, on a donc 14 items.
    La syntaxe générale de ce type de réponse est :

```
\answer{Commentaire}{7,10;\enonce}{type=mark}
```

```
\answer{question 3}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=mark}{option=
\answer{question 4}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=mark}{option=
```

Les options possibles proposent donc plusieurs notations :

- `color` : Cette option permet de spécifier la couleur du surligneur avec un code html ou les couleurs standard définies en html5.
- `split` : Note partielle à la question. Par défaut 1 mauvais choix est compensé par 2 bons. Une telle politique est nécessaire, entre autres, pour dissuader de prendre tous les choix disponibles dans le cas où beaucoup d'entre eux sont bons.
- `eqweight` : Score partiel équivalent. Un bon choix est compensé par un mauvais. Cette politique en apparence juste peut conduire à des effets pervers !

- **Cliquer sur une image ou un texte** (nom : `click`)

Le type `click` compare l'objet cliqué par l'utilisateur avec la bonnes réponse définie par l'enseignnant. Il ne peut être utilisé que dans le cas où il n'y a pas d'autres réponses ou choix car il envoie directement la réponse après le click.
La syntaxe générale est :

```
\answer{Commentaire}{1;juste1,faux1,faux2}{type=click}
\answer{Commentaire}{1;<img src="\url1" alt="Image bonne" />,<img src="\url2" alt="Image error
```

Vous pouvez utiliser l'option `shuffle` pour changer l'ordre dans lequel les choix sont montrés.

```
\answer{Commentaire}{1;juste1,faux1,faux2}{type=click}{option=shuffle}
\answer{Commentaire}{1;<img src="\url1" alt="Image bonne" />,<img src="\url2" alt="Image error
```

Pour mixer les réponses sans l'option shuffle, on doit réaliser au préalable un petit travail sur les variables :

```
\text{listegraph=<img src="\urlL" alt="Repère en L" />,<img src="\urlT" alt="Repère en T inver
\text{li=shuffle(3)}
\text{listegraphmixe=\listegraph[\li]}
\integer{rep=position(1,\li)}

\answer{}{\rep;\listegraphmixe}{type=click}
```

- **Sélectionner une ou plusieurs cellules dans un tableau** (nom : `multipleclick`)

Le type `multipleclick` analyse la ou les cellules sélectionnées par l'élève et les compare aux bonnes réponses proposées par l'enseignant.
La syntaxe générale de ce type de réponse est :

```
\answer{Commentaire}{numero_des_bons_choix;liste_de_choix}{type=multipleclick}
```

Les choix sont mis dans un tableau si cela est indiqué dans le deuxième item de `\embed{}` :

```
\statement{
\embed{r1,X x Y
2 x 3
[a,b;1,2,3]}
}
\answer{}{1,2;a,b,c,d,e,f}{type=multipleclick}
```

- la première ligne contient la dimension du tableau ;
- la deuxième ligne est le nombre de colonnes et de lignes : dans l'exemple ci-dessus, les objets seront disposés dans un tableau ayant 2 colonnes et 3 lignes ;
- Il est possible de rajouter sur la troisième ligne la liste des en-têtes des lignes et des colonnes du tableau sous la forme [nom_des_colonnes;nom_des_lignes]. Si la réponse proposée est bonne, les en-têtes seront vertes sinon rouges.
- La classe css du tableau est `ans_multipleclikn` (avec n le numéro de la question) et peut être configurée pour l'exercice dans le champ `\css{ }`, par exemple,

```
\css{<style type="text/css">
.ans_multipleclick1 {margin-left:auto;margin-right:auto;
border: medium solid #393b40;
border-collapse: separate;
text-align:center;}
td, th {
border: thin solid #393b40;
background-color: #e6ebff;}
</style>}
\matrix{tableau=1,2,3,4,5,6,7,8
205,252,327,349,412,423,441,472
3700,1240,435,150,43,18,7,2
54,110,198,420,780,1640,3160,5960
310,396,425,480,612,704,830,946
3540,3260,3120,2890,2710,2570,2280,2090
509,925,1348,1678,2087,2506,2898,3323}
\text{tableau_en_ligne=semaine,\tableau[1;],Lucas,\tableau[2;],Louis,\tableau[3;],Mathild
\text{numero_des_bons_choix=10,11,12,13,14,15,16,17,18}
\statement{
 \embed{r1,600x400
9 x 7
[a,b,c,d,e,f,g,h,i;1,2,3,4,5,6,7]}
}
\answer{Commentaire}{\numero_des_bons_choix;\tableau_en_ligne}{type=multipleclick}
```

Les options possibles proposent plusieurs notations :

- `split` : Note partielle à la question. Par défaut 1 mauvais choix est compensé par 2 bons. Une telle politique est nécessaire, entre

autres, pour dissuader de prendre tous les choix disponibles dans le cas où beaucoup d'entre eux sont bons.
- `eqweight` : Score partiel équivalent. Un bon choix est compensé par un mauvais. Cette politique en apparence juste peut conduire à des effets pervers !

- **Sélectionner des cartes** (nom : `flashcard`)
  Le type `flashcard` analyse la ou les cartes retournées par l'élève et compare aux bonnes réponses. La note est 0 dès que sa réponse ne s'accorde pas exactement à la réponse attendue.
  La syntaxe générale de ce type de réponse est :

```
\answer{Commentaire}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=flashcard}
\answer{question 2}{1,4;juste1,faux1,très faux,juste2,faux2,faux3}{type=flashcard}{option=shuf
\answer{question 3}{1,4,7;juste1,faux1,très faux,juste2,faux2,faux3,juste3}{type=flashcard}{op
```

  L'élève peut sélectionner une ou plusieurs réponses. Les options possibles sont :
  - `shuffle` : Ordre des réponses aléatoires ;
  - `sort` : Réponses dans l'ordre alphabétique ;
  - `show` : Les cartes sont retournées par défaut et l'élève doit mettre les bons choix face cachée. Le nombre de clics sur chaque carte est renvoyé dans la deuxième ligne de la variable `\reply` n.

  Il est impératif d'insérer le type de réponse `flashcard` dans la commande `\embed` du statement. On peut préciser sur la deuxième et troisième ligne le style css des cartes (style associé à la balise div). Par exemple,

```
\embed{r1,4
style="display:inline-block;min-width:30px;min-height:30px;padding-top:10px;padding-bottom:5p
style="display:inline-block;min-width:30px;min-height:30px;padding-top:10px;padding-bottom:5p
}
```

Types spécifiques ou avancés

- **Formule brute d'une molécule** (nom : `chembrut`) permet de vérifier une formule brute. Elle est partiellement bonne si elle n'est pas écrite dans l'ordre (carbone, hydrogène, puis les autres éléments par ordre alphabétique). La bonne réponse doit être sous la forme C7H4Cl2O2. La variable `reply` i contient sur la première ligne la réponse, puis la réponse formatée sous la forme

```
C,7
H,4
Cl,2
O,2
```

  dans l'ordre donné par la réponse. La variable `sc_reply` i vaut 0.5 si la réponse est correcte à l'ordre près des éléments
- **Dessiner une molécule** (nom : `chemdraw`) Une applet (modification par Joke Evers de l'applet SketchEl http://sketchel.sourceforge.net) permettant de dessiner une molécule est affichée.

  La réponse est définie comme l'adresse d'un fichier du type MDMol (extension .mol) correspondant à la molécule ou son contenu (voir plus loin pour l'adresse). D'autres types de fichiers pourront être reconnus ultérieurement. Ce fichier peut par exemple obtenu préliminairement grâce à l'applet (fonctionnant seul) ou être un fichier mol conforme (version 2000). Dans l'avenir, une banque de tels fichiers sera accessible dans wims.

  La molécule dessinée est analysée et comparée grâce aux programmes `checkmol` et `matchmol` (http://merian.pch.univie.ac.at/~nhaider/cheminf/cmmm.html) qui se trouve maintenant sur le serveur WIMS (à partir de la version 3.65a).

  La variable `\reply` i est une matrice (avec ; comme séparateur de ligne)

  - dans le cas d'une mauvaise réponse,
    - Ligne 1 : les caractéristiques de la réponse (voir plus loin)
    - Ligne 2 : les fonctions caractéristiques de la réponse (voir plus loin)
    - Ligne 3 : la formule brute de la réponse
    - Ligne 4 : les caractéristiques de la bonne réponse (voir plus loin)
    - Ligne 5 : les fonctions caractéristiques de la bonne réponse (voir plus loin)
    - Ligne 6 : pour l'instant, vide
    - Ligne 7 : la liste des numéros des lignes pour lesquels les deux molécules diffèrent selon les sorties de matchmol (voir plus loin)
    - Lignes suivantes : un feedback prérempli correspondant à chacune des numéros de lignes pour lesquels les deux molécules diffèrent (dans le même ordre que les numéros de la ligne 1).
  - dans le cas d'une bonne réponse (ce qui peut se tester par la valeur 1 de la variable `\sc_reply` i), les quatre premières lignes uniquement.
  - lorsqu'aucune bonne réponse n'est fournie, les lignes 1 à 3.

  Il est possible que soient rajoutés ultérieurement d'autres items dans les lignes 3 et 7, il est donc conseillé si elles sont utilisées de prendre systématiquement le premier item de ces lignes.

Options disponibles :

Nom	Exemple	Explications
`image=adresse`	`image=ch/ethane.png`	adresse de l'image à partir de la valeur de **`\imagedir`** défini dans **`var.proc`** , dans l'exemple l'image est dans le répertoire **`\imagedir/ch`**
match= `matchmol_option`	`match=gG`	`g` : check geometry of double bonds (E/Z) `G` : check geometry of chiral centers (R/S) `a` : check charges strict `i` : check isotopes strict `d` : check radicals strict
`\answer{}{}{type=chemdraw}{option=image=ch/ethane.png match=gG}`		

  Les options du champ `embed` sont la taille `L x H` sur la première ligne suivie sur une nouvelle ligne des options possibles de l'applet (valeurs yes (ou 1) et no (ou 0))

  Une option `help` est disponible, elle rajoute automatiquement une aide sur l'applet (il est possible de n'en prendre qu'une partie en choisissant des mots clés parmi `single,atom,double,erasor,template` (par exemple `help=[single,erasor]`).

Nom	Défaut	Explications
Menu (barre horizontale en haut)		
menu_block	1	menu d'édition
menu_select	1	permet de sélectionner un atome, de passer d'un atome à l'autre
menu_transform	0	transformations (rotations, symétrie) et utilisation des modèles
menu_zoom	1	permet de zoomer
menu_show	0	permet de faire apparaître ou disparaître les carbones etc
menu_hydrogen	0	permet de faire apparaître ou non les hydrogènes
menu_stereo	0	Menu de stéréochimie
Outils (barre verticale à gauche)		
tool_cursor	1	indispensable, permet de sélectionner une zone
tool_rotator	0	permet la rotation des molécules
tool_erasor	1	permet d'effacer
tool_dialog	0	
tool_edit	0	permet d'écrire le nom des molécules (texte libre)
tool_setatom	1	propose une liste d'atomes (la liste peut être précisée dans le paramètres `atoms`
tool_single	1	liasion simple
tool_double	1	liaison double
tool_triple	1	liaison triple
tool_zero	0	liaison nulle
tool_inclined	0	liaison inclinée
tool_declined	0	liaison déclinée
tool_unknown	0	liaison inconnue
tool_charge	0	charges
tool_undo	1	permet de revenir en arrière
tool_redo	1	permet de refaire
tool_template	1	outil donnant des modèles, par défaut, ce sont les modèles de l'applet ; on peut configurer quels modèles apparaîtront en donnant des valeurs à `template` (séparés par un espace ...)
tool_cut	0	Couper
tool_copy	0	Copier
tool_paste	0	Coller
tool_select	0	Sélectionner
tool_unselect	0	Déselctionner
Autres options		
atoms	liste (entre guillemets) d'atomes séparés par des espaces, par exemple `atoms="C N O S Mn Ni Mg Br Cl I"`	
template	liste des noms des molécules à mettre dans le template, si le mot cyclo s'y trouve, met les cycles de base)	
file	adresse du fichier MDMol, ce fichier doit se trouver dans le répertoire **`data`** du module (adresse de la forme **`data/xxx`**), dans le répertoire **`data`** de la distribution (adresse de la forme **`data/xxx`**) ou dans un module de données (adresse de la forme `datamodule/xxxx`))	
show_carbon	montrer les atomes de carbone	
show_hydrogen	montrer les atomes d'hydrogènes	

Le tableau suivant indique la signification des différents items des lignes 1 et 4 de `\reply` i (selon la documentation de `checkmol`)

Numéro	Nom	
1	n_atoms	number of heavy atoms
2	n_bonds	number of bonds between non-H atoms
3	n_rings	number of rings
4	n_QA	number of query atoms
5	n_QB	number of query bonds
6	n_chg	number of charges
7	n_C1	number of sp-hybridized carbon atoms
8	n_C2	number of sp2-hybridized carbon atoms
9	n_C	total number of carbon atoms
10	n_CHB1p	number of carbon atoms with at least 1 bond to a hetero atom
11	n_CHB2p	number of carbon atoms with at least 2 bonds to a hetero atom
12	n_CHB3p	number of carbon atoms with at least 3 bonds to a hetero atom

13	n_CHB4	number of carbon atoms with 4 bonds to a hetero atom
14	n_O2	number of sp2-hybridized oxygen atoms
15	n_O3	number of sp3-hybridized oxygen atoms
16	n_N1	number of sp-hybridized nitrogen atoms
17	n_N2	number of sp2-hybridized nitrogen atoms
18	n_N3	number of sp3-hybridized nitrogen atoms
19	n_S	number of sulfur atoms
20	n_SeTe	total number of selenium and tellurium atoms
21	n_F	number of fluorine atoms
22	n_Cl	number of chlorine atoms
23	n_Br	number of bromine atoms
24	n_I	number of iodine atoms
25	n_P	number of phosphorus atoms
26	n_B	number of boron atoms
27	n_Met	total number of metal atoms
28	n_X	total number of "other" atoms (not listed above) and halogens
29	n_b1	number of single bonds
30	n_b2	number of double bonds
31	n_b3	number of triple bonds
32	n_bar	number of aromatic bonds
33	n_C1O	number of C-O single bonds
34	n_C2O	number of C=O double bonds
35	n_CN	number of C/N bonds (any type)
36	n_XY	number of heteroatom/heteroatom bonds (any type)
37	n_r3	number of 3-membered rings
38	n_r4	number of 4-membered rings
39	n_r5	number of 5-membered rings
40	n_r6	number of 6-membered rings
41	n_r7	number of 7-membered rings
42	n_r8	number of 8-membered rings
43	n_r9	number of 9-membered rings
44	n_r10	number of 10-membered rings
45	n_r11	number of 11-membered rings
46	n_r12	number of 12-membered rings
47	n_r13p	number of 13-membered or larger rings
48	n_rN	number of rings containing nitrogen (any number)
49	n_rN1	number of rings containing 1 nitrogen atom
50	n_rN2	number of rings containing 2 nitrogen atoms
51	n_rN3p	number of rings containing 3 or more nitrogen atoms
52	n_rO	number of rings containing oxygen (any number)
53	n_rO1	number of rings containing 1 oxygen atom
54	n_rO2p	number of rings containing 2 or more oxygen atoms
55	n_rS	number of rings containing sulfur (any number)
56	n_rX	number of heterocycles (any type)
57	n_rar	number of aromatic rings (any type)
58	n_rbz	number of benzene rings
59	n_br2p	number of bonds belonging to two or more rings
60	n_psg01	number of atoms belonging to group 1 of the periodic system
61	n_psg02	number of atoms belonging to group 2 of the periodic system
62	n_psg13	number of atoms belonging to group 13 of the periodic system
63	n_psg14	number of atoms belonging to group 14 of the periodic system
64	n_psg15	number of atoms belonging to group 15 of the periodic system
65	n_psg16	number of atoms belonging to group 16 of the periodic system
66	n_psg17	number of atoms belonging to group 17 of the periodic system
67	n_psg18	number of atoms belonging to group 18 of the periodic system

68	n_pstm	number of atoms belonging to the transition metals
69	n_psla	number of atoms belonging to the lanthanides or actinides
70	n_iso	number of isotopes
71	n_rad	number of radicals

Le tableau suivant donne l'explication des codes (lignes 2 et 5 de `reply i`). On peut alors à l'aide de conditions tester les fonctions présentes dans la molécule tracée.

Code	Explication
000000T2	cation
000000T1	anion
C2O10000	carbonyl
C2O1H000	aldehyde, aldéhyde
C2O1C000	ketone, cétone
C2S10000	thiocarbonyl
C2S1H000	thioaldehyde
C2S1C000	thioketone
C2N10000	imine, imine
C2N1N000	hydrazone, hydrazone
C2NNC4ON	semicarbazone
C2NNC4SN	thiosemicarbazone
C2N1OH00	oxime, oxime
C2N1OC00	oxime_ether, oxime éther
C3OC0000	ketene, cétène
C3OCC000	ketene_acetal_deriv
C2O2H200	carbonyl_hydrate
C2O2HC00	hemiacetal, hémiacetal
C2O2CC00	acetal, acétal
C2NOHC10	hemiaminal, hémiaminal
C2N2CC10	aminal, aminal
C2NSHC10	thiohemiaminal
C2S2CC00	thioacetal, thioacétal
C2CNH000	enamine, énamine
C2COH000	enol, énol
C2COC000	enolether, éther d'énol
O1H00000	hydroxy && hydroxy_generic, groupement hydroxy
O1H0C000	alcohol, alcool
O1H1C000	prim_alcohol, alcool primaire
O1H2C000	sec_alcohol, alcool secondaire
O1H3C000	tert_alcohol, alcool tertiaire
O1H0CO1H	1_2_diol, diol-1, 2
O1H0CN1C	1_2_aminoalcohol, 1_2_aminoalcohol
O1H1A000	phenol, phénol
O1H2A000	1_2_diphenol, diphenol-1, 2
C2COH200	enediol
O1C00000	ether && ether_generic, éther
O1C0CC00	dialkylether, dialkyléther
O1C0CA00	alkylarylether, alkylaryéther
O1C0AA00	diarylether, diaryéther
S1C00000	thioether, thioéther
S1S1C000	disulfide, disulfure
O1O1C000	peroxide, peroxyde
O1O1H000	hydroperoxide, hydroperoxyde
N1N10000	hydrazine, hydrazine
N1O1H000	hydroxylamine, hydroxylamine
N1C00000	amine, amine
N1C10000	prim_amine, amine primaire

N1C1C000	prim_aliph_amine, amine p-primaire
N1C1A000	prim_arom_amine, amine p-primaire aromatique
N1C20000	sec_amine, amine secondaire
N1C2CC00	sec_aliph_amine
N1C2AC00	sec_mixed_amine
N1C2AA00	sec_arom_amine
N1C30000	tert_amine, amine tertiaire
N1C3CC00	tert_aliph_amine
N1C3AC00	tert_mixed_amine
N1C3AA00	tert_arom_amine
N1C400T2	quart_ammonium, sel d'amonium quaternaire
N0O10000	n_oxide, N-oxyde
XX000000	halogen_deriv, dérivé halogéné
XX00C000	alkyl_halide
XF00C000	alkyl_fluoride
XC00C000	alkyl_chloride
XB00C000	alkyl_bromide
XI00C000	alkyl_iodide
XX00A000	aryl_halide
XF00A000	aryl_fluoride
XC00A000	aryl_chloride
XB00A000	aryl_bromide
XI00A000	aryl_iodide
000000MX	organometallic
000000ML	organolithium, organolithien
000000MM	organomagnesium, organomagnesien
C3O20000	carboxylic_acid_deriv
C3O2H000	carboxylic_acid, acide carboxylique
C3O200T1	carboxylic_acid_salt, carboxylate
C3O2C000	carboxylic_acid_ester, ester
C3O2CZ00	lactone, lactone
C3ONC000	carboxylic_acid_amide, amide
C3ONC100	carboxylic_acid_prim_amide, amide primaire
C3ONC200	carboxylic_acid_sec_amide, amide secondaire
C3ONC300	carboxylic_acid_tert_amide, amide tertiaire
C3ONCZ00	lactam, lactame
C3ONN100	carboxylic_acid_hydrazide
C3ONN200	carboxylic_acid_azide
C3ONOH00	hydroxamic_acid
C3N2H000	carboxylic_acid_amidine
C3NNN100	carboxylic_acid_amidrazone
C3N00000	nitrile, nitrile
C3OXX000	acyl_halide
C3OXF000	acyl_fluoride, fluorure d'acide|fluorure d'acyle
C3OXC000	acyl_chloride, chlorure d'acide|chlorure d'acyle
C3OXB000	acyl_bromide, bromure d'acide|bromure d'acyle
C3OXI000	acyl_iodide, iodure d'acide|iodure d'acyle
C2OC3N00	acyl_cyanide, cyanure d'acide|cyanure d'acyle
C3NOC000	imido_ester
C3NXX000	imidoyl_halide
C3SO0000	thiocarboxylic_acid_deriv
C3SOH000	thiocarboxylic_acid, thioacide|thioacide carboxylique
C3SOC000	thiocarboxylic_acid_ester
C3SOCZ00	thiolactone, thiolactone
C3SNH000	thiocarboxylic_acid_amide

C3SNCZ00	thiolactam, thiolactame
C3NSC000	imido_thioester
C3ONAZ00	oxohetarene
C3SNAZ00	thioxohetarene
C3NNAZ00	iminohetarene
C3O30000	orthocarboxylic_acid_deriv
C3O3C000	carboxylic_acid_orthoester, orthoester
C3O3NC00	carboxylic_acid_amide_acetal
C3O2C3O2	carboxylic_acid_anhydride, anhydride d'acide
C3ONC000	carboxylic_acid_imide, imide
C3ONCH10	carboxylic_acid_unsubst_imide, imide
C3ONCC10	carboxylic_acid_subst_imide, imide
C4000000	co2_deriv
C4O30000	carbonic_acid_deriv
C4O3C100	carbonic_acid_monoester
C4O3C200	carbonic_acid_diester, carbonate
C4O3CX00	carbonic_acid_ester_halide
C4SO0000	thiocarbonic_acid_deriv
C4SOC100	thiocarbonic_acid_monoester
C4SOC200	thiocarbonic_acid_diester
C4SOX_00	thiocarbonic_acid_ester_halide
C4O2N000	carbamic_acid_deriv
C4O2NH00	carbamic_acid, acide caramique
C4O2NC00	carbamic_acid_ester, carbamate
C4O2NX00	carbamic_acid_halide
C4SN0000	thiocarbamic_acid_deriv
C4SNOH00	thiocarbamic_acid
C4SNOC00	thiocarbamic_acid_ester
C4SNXX00	thiocarbamic_acid_halide
C4O1N200	urea, urée
C4N2O100	isourea
C4S1N200	thiourea, thiourée
C4N2S100	isothiourea
C4N30000	guanidine, guanidine
C4ON2N00	semicarbazide
C4SN2N00	thiosemicarbazide
N4N20000	azide, azoture
N2N10000	azo_compound, hydrazone
N3N100T2	diazonium_salt, sel diazonium
N3C10000	isonitrile, isonitrile
C4NO1000	cyanate, cyanate
C4NO2000	isocyanate, isocyanate
C4NS1000	thiocyanate, thyiocyanate
C4NS2000	isothiocyanate, isothiocyanate
C4N20000	carbodiimide, carbodiimide
N2O10000	nitroso_compound, nitroso
N4O20000	nitro_compound, nitro
N3O20000	nitrite, nitrite
N4O30000	nitrate, nitrate
S6O00000	sulfuric_acid_deriv
S6O4H000	sulfuric_acid, acide sulfurique
S6O4HC00	sulfuric_acid_monoester
S6O4CC00	sulfuric_acid_diester
S6O3NC00	sulfuric_acid_amide_ester
S6O3N100	sulfuric_acid_amide

S6O2N200	sulfuric_acid_diamide
S6O3XX00	sulfuryl_halide
S5O00000	sulfonic_acid_deriv
S5O3H000	sulfonic_acid, acide sulfonique
S5O3C000	sulfonic_acid_ester
S5O2N000	sulfonamide, sulfonamide
S5O2XX00	sulfonyl_halide, halogénure de sulfonyle
S4O20000	sulfone, sulfone
S2O10000	sulfoxide, sulfoxyde
S3O00000	sulfinic_acid_deriv
S3O2H000	sulfinic_acid
S3O2C000	sulfinic_acid_ester
S3O1XX00	sulfinic_acid_halide
S3O1N000	sulfinic_acid_amide
S1O00000	sulfenic_acid_deriv
S1O1H000	sulfenic_acid
S1O1C000	sulfenic_acid_ester
S1O0XX00	sulfenic_acid_halide
S1O0N100	sulfenic_acid_amide
S1H10000	thiol, thiol
S1H1C000	alkylthiol, thiol
S1H1A000	arylthiol, thiophénol
P5O0H000	phosphoric_acid_deriv
P5O4H200	phosphoric_acid, acid phosphorique
P5O4HC00	phosphoric_acid_ester
P5O3HX00	phosphoric_acid_halide
P5O3HN00	phosphoric_acid_amide
P5O0S000	thiophosphoric_acid_deriv
P5O3SH00	thiophosphoric_acid
P5O3SC00	thiophosphoric_acid_ester
P5O2SX00	thiophosphoric_acid_halide
P5O2SN00	thiophosphoric_acid_amide
P4O30000	phosphonic_acid_deriv
P4O3H000	phosphonic_acid, acid phosphonique
P4O3C000	phosphonic_acid_ester
P3000000	phosphine, phosphine
P2O00000	phosphinoxide, oxyde de phosphine
B2O20000	boronic_acid_deriv
B2O2H000	boronic_acid, acide boronique
B2O2C000	boronic_acid_ester, ester boronique
000C2C00	alkene, alcène
000C3C00	alkyne, alcyne
0000A000	aromatic, aromatique
0000CZ00	heterocycle, composé hétérocyclique
C3O2HN1C	alpha_aminoacid, alpha_aminoacide
C3O2HO1H	alpha_hydroxyacid, alpha-hydroxyacide

-s strict comparison of atom and bond types (including ring check) -r force SSR (set of small rings) ring search mode -g check geometry of double bonds (E/Z) -G check geometry of chiral centers (R/S) -a check charges strict -i check isotopes strict -d check radicals strict xaigdGs

- **Clic sur des atomes d'une molécule en 2D** (nécessite java) (nom : `chemclick`) Une molécule est présentée dans l'applet WIMSChem et on doit cliquer sur certains atomes ou certaines liaisons. Ce type de réponse doit être utilisé avec la commande `\embed{}`. La première ligne de la commande `\embed` est formée de la taille en pixels de l'applet X x Y. Sur la deuxième ligne, des mots d'option peuvent être mis (voir type chemdraw). La bonne réponse est :
  - Première ligne. la liste des numéros des atomes et/ou des numéros des liaisons dans le fichier MDMol sous la forme `atoms:1,2 bonds:4,1`.
  - Deuxième ligne. L'adresse d'un fichier MDMol (extension `.mol`) décrivant la molécule. Ce fichier doit être dans le répertoire **`data`** du module, dans le répertoire data de la distribution wims (adresse **`data/xxx`**) dans un module de données qui se trouve dans le répertoire modules/data de WIMS (dans ce cas, l'adresse est de la forme datamodule/xxx) ou dans le dossier images du module (adresse **`\imagedir/xxx`**). Il est aussi possible de mettre le contenu du fichier.

  Les atomes sélectionnés par click droit sont colorés de manière différente. La variable `\reply`n est de la forme `atoms:1,2 bonds:2` Une option `help` est possible, elle rajoute une aide sur l'applet (pour l'instant uniquement sur l'icone de sélection).

```
\answer{}{atoms:1,2 bonds:1;data/file.mol}{type=chemclick}
```

- **Équation chimique.** (nom : `chemeq`)
  C'est une entrée texte à comparer à une équation chimique donnée, la syntaxe étant définie par l'analyseur *chemeq*. On définit la réponse comme une équation chimique modèle (équilibrée), éventuellement suivie par un point-virgule et l'équation (peut-être) non équilibrée à proposer au début.

  La réponse est considérée comme correcte si les nombres stoechiométriques sont proportionnels au modèle. L'ordre des composés dans chaque membre n'a pas d'importance, et la syntaxe autorise de nombreuses variantes, dont des coefficients fractionnaires.

  Exemple:
  - **2H2 + O2 -> 2H2O; H2 + O2 -> H2O** le modèle est l'équation de la synthèse de l'eau, et un exemple non équilibré est donné au départ.
- **Ensemble de caractères.** (name: `chset`)
  C'est une liste de caractères alpha-numériques où l'ordre n'a pas d'importance, ni majuscules-minuscules.

  Il peut être utilisé pour faire des réponses à choix multiples, où le nombre de choix présenté doit être fixé, et où il peut y avoir plusieurs bons choix que l'utilisateur doit trouver tous.

  Si le mot `norepeat` est présent dans l'option, la répétition d'un même caractère est ignoré par le comparateur.

- **Champs à remplir avec glisser-déposer.** (noms : `clickfill` et `dragfill`)

  Le champ de réponse, habituellement inséré dans l'énoncé, peut être rempli en cliquant sur l'objet donné (texte, formule ou image) ou en le glissant à la souris depuis une liste d'objets.

  La réponse est habituellement une matrice dont la première ligne est la bonne réponse et la deuxième est une liste d'items qui sont les mauvaises réponses.

  La bonne réponse peut être une liste de plusieurs items. Dans ce cas, le champ à remplir aura une longueur L (le nombre d'objets qu'il peut contenir) plus grande que 1.

  On peut définir des bonnes réponses multiples dans la première ligne, en les séparant par le caractère `|`. Et la réponse peut être analysée si la première ligne est une variable non définie.

  Il est conseillé d'utiliser ce champ de réponse via `\embed{}`, avec une taille de champ de réponse sous la forme `H x V x L x T`, où `H` et `V` sont respectivement les tailles horizontale et verticale (en pixels) d'une étiquette, où `L` est le nombre d'étiquettes à remplir dans le champ de réponse et où `T` est le nombre de lignes. Par défaut, `L` est le nombre d'objets de la réponse et `T` vaut 1, autrement dit le champ de réponse est sur une seule ligne. Si `T` est égal à `L`, le champ de réponse se présentera comme une liste verticale.

  Il peut y avoir plusieurs réponses de type `clickfill` ou `dragfill` dans un exercice, mais elles doivent être de même type (toutes `clickfill` ou toutes `dragfill`), et les étiquettes doivent avoir la même taille. Les objets sont alors mélangés.

  Les types `clickfill` et `dragfill` ont des interfaces utilisateur très semblables (les deux peuvent être cliquées et glissées). La différence principale est qu'un objet peut être utilisé plusieurs fois pour le type `clickfill` et au plus une fois pour le type `dragfill`.

  Le contenu des étiquettes est par défaut centré. Il est possible d'aligner le contenu sur la gauche ou sur la droite en utilisant l'option `align=left` ou `align=right` (`{option= align=left}`).

  Si on utilise l'option `noorder`, l'analyse de la réponse ne tient pas compte de l'ordre dans lequel les étiquettes ont été déposées. Cela permet des exercices du type *classer par propriétés*.

  Avec l'option `transparent`, le champ à remplir est transparent (peut être utile avec la méthode spéciale `imagefill`).

  Avec l'option `keeporder`, les étiquettes sont présentées dans l'ordre de la liste.

  Ces types, fondés sur DynAPI, peuvent ne pas fonctionner avec certains navigateurs.

  Exemple où l'on a plusieurs lignes possibles :

```
\statement{ Donner la transposée de la matrice \([1,2,3;4,5,6;7,8,9]\)
<div class="wimscenter">\embed{r1, 50x50x9x3}</div>
}
\answer{}{1,4,7,2,5,8,3,6,9}{type=dragfill}
```

- **Cliquer sur des pavés.** (nom : `clicktile`)
  Une applet affiche un pavage par des pavés rectangulaires de couleur. Il est possible de tracer de plus des lignes polygonales ou (exclusif) un point. Il est demandé de mettre dans des couleurs différentes certains des pavés.

  Ce type de réponse nécessite l'utilisation de la commande `\embed`. Chaque carré est repéré par son coin en haut à gauche sous la forme `entier:entier`. La première ligne de la commande `\embed` est formée de la taille en pixels de l'applet X x Y. La seconde ligne entre `[` et de `]` est formée des commandes précisant les pavés colorés (ils ne pourront pas être modifiés) : par exemple

```
\embed{r1, 200 x 200
[xrange -5,5
yrange -5,5
background_color yellow
square blue,1:1,1:2,1:3
square green,2:3,2:4
point red,0:0]}
```

  Les bornes (`xrange` et `yrange`) doivent être entières. La bonne réponse est formée de lignes de la forme `color,a:b,c:d` Les points (`a:b`) repérant les carrés doivent avoir des coordonnées entières. Pour le dessin supplémentaire, l'une des lignes est possible

```
polygon white,0:0,0:1,1:1
point red,0:0
line white,-5:0,5:0
```

  Il est possible de mettre une image en arrière-plan. Si elle est faite avec flydraw, il faut rajouter le résultat de la fonction `draw()`. Il est recommandé de prendre le même `xrange` et `yrange`. La variable `reply` n contient la matrice des carrés sélectionnés (une ligne par couleur). Si la ligne

```
return_all_objects yes
```

est dans la deuxième ligne de la commande `\embed`, tous les carrés seront retournés dans la réponse. La variable `sc_reply` n est égale à 0.5 si tous les carrés cliqués sont corrects mais de mauvaise couleur.

Il est possible de mettre une variable vide dans la réponse. Dans ce cas, l'analyse doit être faite par des conditions.

Exemple :

```
\statement{
 \embed{r1,200 x 200
 [xrange -5,5
  yrange -5,5
  background_color yellow
  square blue,1:1,1:2,1:3
  square green,2:3,2:4
  line white,-5:0,5:0
 ]}
}
```

	Explication
xrange x1,x2	
yrange y1,y2	
background_color [color]	
square [color],x1:y1,x2:y2,...	
segment [color],x1:y1,x2:y2	
polygon [color],x1:y1,x2:y2,x2:y2,...	
return_all_objects	no : seuls les nouveaux objets sont retournés
colors	Liste de couleurs supplémentaires

Les couleurs possibles sont `white,red, green, blue, orange, yellow, purple, lightgreen, lightblue, cyan, brown, salmon, pink.`

- **Horloge.** (nom: `clock`)
Réglage de l'heure sur une horloge. La réponse est donnée sous la forme `h:min:sec` où `h` désigne les heures, `min` les minutes et S `sec` les secondes.

Exemple simple

```
\statement{Il est 9h 39 min et 31 secondes
\embed{r1,200x200}
}
\answer{}{9:39:31}{type=clock}
```

Options

```
1 \statement{Il est 9h 30 min et 35 secondes
2 \embed{r1,200x200}
3 }
4 \answer{}{9:30:35}{type=clock}{option=init=[5,45,5] button=[1,5,5] clocktype=0}
5
```

Éditeur actif

	Explications		Défaut
`clocktype`	Type de l'horloge	0: avec graduation, 1: avec chiffres, 2: avec graduation	2
`color`	`[H,M,S,B,F]` Personnalisation des couleurs de l'horloge	Il peut y avoir jusqu'à 5 couleurs : dans l'ordre, `H` : aiguille des heures, `M` : aiguille des minutes, `S` : aiguille des secondes, `B` : fond, `F` : chiffres	[black,black,red,white,black]
`init`	`[h,m,s]` Initialisation de l'horloge	`h` valeur des heures, `m` valeur des minutes, `s` valeur des secondes	[0,0,0]
`button`	`[a,b,c]` Paramétrisation des boutons	l'aiguille des heures avance de `a` en `a` heures, l'aiguille des minutes avance de `b` en `b` minutes, l'aiguille des secondes avance de `c` en `c` secondes.	[1,1,1]

- **Phrase composée d'éléments donnés.** (nom : `compose`)

Les éléments donnés peuvent être du texte (mots ou groupes de mots), des formules (formatées par \(...)), ou des images. Ils doivent être listés en items dans la bonne réponse fournie. Des éléments inutiles peuvent être ajoutés dans la réponse après un point-virgule `;'.

You may define synonymes in the good answer. To do this, the synonymes has only to be added behind the (standard) good answer, preceeded by the vertical bar `|'. An arbitrary number of synonymes can be defined. In such a case, only the first synonyme (before the first `|') should have comma-separated items. Words not appearing in the first synonyme will not be presented, unless added after a semi-colon.

On peut mettre en option le séparateur qui apparaitra entre les textes : linkword="mot de liaison". Par défaut, il s'agit d'un espace.

Ce type, fondé sur DynAPI, peut ne pas fonctionner avec certains navigateurs.

Remarque. Ce type de réponse accepte l'option symtext. Si le mot `symtext` est déclaré dans l'option de la réponse, l'analyse de la réponse sera exactement comme pour symtext, en particulier sans aucun traitement préalable des textes. Et toutes les options symtext seront comprises dans ce cas.

- **Nombre complexe.** (nom: `complex`)
Comparaison de nombres complexes à la précision définie par l'auteur.

- Si le mot `comma` est présent dans l'option, l'écriture des nombres décimaux avec une virgule décimale est acceptée (mais la bonne réponse doit être un nombre avec le point décimal). Si la réponse est mauvaise, la bonne réponse est donnée avec le même "séparateur" décimal que celui donné par l'élève.
- La précision peut être précisée dans le champ `precision`, par défaut 1000.

  L'erreur est par défaut relative : la différence doit être de module inférieur ou égal à `(good_answer+reply)/precision` (remplacer `precision` par `sqrt(precision)` pour une réponse partiellement bonne).

  Si le mot `absolute` est un mot d'option, la précision est absolue. La réponse est alors partiellement bonne si la différence est inférieure à `10/precision`.

- Si la lettre `j` est présente dans l'option, les réponses du type `2+7*j` sont acceptées en même temps que les réponses du type `2+7*i`. Si la réponse est mauvaise, la bonne réponse est donnée avec la même lettre que celle donnée par l'élève. Par contre, le programmeur doit utiliser la lettre `i` pour les calculs préliminaires et l'écriture de la bonne réponse.

- **Clic sur une image.** (nom : `coord`)
  Ce type de réponse permet à l'utilisateur de cliquer sur une image, et peut analyser la position du clic suivant divers critères de zone.

  La bonne réponse doit être donnée sous la forme d'une matrice (le séparateur de lignes est le point-virgule `;'). La première ligne est l'URL de l'image. La seconde ligne est la position des critères permettant de considérer les coordonnées du clic comme correctes. Et à partir de la troisième ligne éventuelle, chacune peut contenir des critères (mauvaises positions). Toutes les coordonnées de points doivent être en pixels.

  La réponse de l'utilisateur est considérée comme bonne lorsque les conditions de la deuxième ligne sont remplies. Sinon, si les critères de mauvaises positions existent (à partir de la 3ème ligne), la réponse est mauvaise si l'une de ces mauvaises conditions est remplie, et ambigue si aucune condition n'est remplie (dans ce dernier cas, l'utilisateur devra réessayer de répondre). S'il n'y a pas de critère de mauvaise position, toute réponse ne remplissant pas les conditions de la deuxième ligne sera jugée mauvaise.

  Le numéro de la première des lignes en accord avec la réponse de l'utilisateur, diminué de 1, est stocké dans une variable `\result`*n*, où *n* est le numéro du champ de réponse et peut être utilisé dans un `feedback`.

  Plusieurs conditions peuvent être combinées dans la même ligne, en utilisant les connecteurs logiques `&' (intersection), `|' (union), `^' (complémentaire), ainsi que les parenthèses.

  Il est recommandé d'insérer ce champ de réponse dans l'énoncé. Il ne peut pas coexister avec d'autres champs de réponse, pour la raison évidente que tout clic de l'utilisateur sur une image enverra le formulaire de réponse.

  L'option `feedback=[ligne 1 de code; ligne 2 de code ; ...]` permet d'insérer une ligne de dessin supplémentaire dans la réponse (attention, cette ligne doit être codée en pixels). Pour des raisons techniques, il est recommandé de mettre au préalable `ligne 1 de code; ligne 2 de code ; ...` dans une variable.

Liste des conditions disponibles pour tester le clic de l'utilisateur

Syntaxe	Signification
`point,x,y`	Point en (x,y). C'est un point ``épais", de largeur fixe.
`rectangle,x1,y1,x2,y2`	L'intérieur d'un rectangle de diagonale (x1,y1)---(x2,y2)
`circle,x,y,d`	L'intérieur d'un cercle de centre (x,y) et diamètre d
`ellipse,x,y,w,h`	L'intérieur d'une ellipse de centre (x,y), largeur w, et hauteur h
`polygon,x1,y1,x2,y2,x3,y3,...`	L'intérieur d'un polygone engendré par les points (x1,y1), (x2,y2), (x3,y3), ..

`bound,NOMFIC,x,y`	zone définie dans le fichier image NOMFIC, qui doit être de la même taille que l'image cliquée (mais peut être une image différente). La condition est remplie si (x,y) est dans la même zone de remplissage que le clic de l'utilisateur. Si le champ `option` de la réponse contient le mot `fill`, la zone est remplie dans la réponse en vert ou en rouge (au lieu d'une boule). Attention. Si l'exercice est dans une classe, l'image NOMFIC doit être recopiée :
	`\text{cache=draw(\size[1],\size[2]`
	`copy 0,0,-1,-1,-1,-1,NOMFIC)}`
	`\text{cache = slib(oef/insfilename)}`
	Dans un module, il suffit d'écrire son nom. Deux options sont possibles dans ce cas :
	• `fill` : colorie la zone plutôt que de mettre une boule dans la réponse (ne pas utiliser dans le cas où l'on utilise une image auxiliaire pour analyser la réponse).
	• `goodanswershown` : indique aussi la bonne réponse sur l'image en cas de réponse fausse.

`bound,NOMFIC`	comme ci-dessus, la condition est remplie si le clic de l'utilisateur a une couleur DIFFERENTE du pixel du coin en haut à gauche de NOMFIC

- **Correspondance bijective d'objets.** (nom : `correspond`)
  Deux ensembles finis d'items, ayant le même nombre d'éléments, sont présentés à l'utilisateur, sous la forme d'une table à deux colonnes, leur correspondance étant mélangée. L'utilisateur peut cliquer sur ces items pour les réordonner jusqu'à obtenir la correspondance correcte.

  La bonne réponse doit être donnée sous forme de matrice à deux lignes. La première ligne est celle des objets présentés à gauche, la seconde celle des objets de droite. Elles doivent être séparées par un point-virgule `;'. Il est possible d'avoir le même item plusieurs fois dans la colonne de droite, mais pas dans la colonne de gauche.

  Les items peuvent contenir du texte, des images (au format HTML), et des formules mathématiques utilisant `\(...)`.

  Il est préférable d'utiliser ce champ de réponse via \embed{}, avec une taille de champ sous la forme `VxHGxHD`, où V est la taille verticale des items, et HG, HD les tailles horizontales des colonnes gauche et droite. Par exemple,

```
\embed{reply1,200 x 100 x 100}
```

  Par défaut, la note 0 est attribuée à toute réponse non entièrement correcte. Si vous ajoutez le mot `split` dans le champ optionnel "option" de la réponse, alors le score sera proportionnel au nombre de bonnes correspondances.

  Ce type, fondé sur DynAPI, peut ne pas fonctionner avec certains navigateurs.

- **Mots croisés.** (nom : `crossword`)

Crée un mot croisé (mots fléchés) à partir d'une matrice de lettres ou de chiffres. (un slib permettant de créer la matrice à partir des mots est prévu).

La bonne réponse est formée de deux items : le premier item est une matrice entre crochets, les coefficients vides correspondant aux cases noires : de la forme

```
[c,i,n,q,,,,
,,,u,n,,,
,,,a,,,s,
,,,t,r,o,i,s
,,,r,,,x,
,,d,e,u,x,,]
```

Le deuxième item est formé des définitions des mots à chercher (sur chaque ligne, le premier item est le mot, le deuxième sa définition). Par exemple

```
[un,1 en lettres
deux,2 en lettres
trois,3 en lettres
quatre,4 en lettres
cinq,5 en lettres
six,6 en lettres]
```

Exemple :

```
\answer{}{[c,i,n,q,,,,
,,,u,n,,,
,,,a,,,s,
,,,t,r,o,i,s
,,,r,,,x,
,,d,e,u,x,,],[un,1 en lettres
deux,2 en lettres
trois,3 en lettres
quatre,4 en lettres
cinq,5 en lettres
six,6 en lettres]}{type=crossword}
```

Les matrices peuvent être écrites avec des points virgules (sans retour à la ligne) :

```
\answer{}{[c,i,n,q,,,,;,,,u,n,,,;,,,a,,,s,;,,,t,r,o,i,s;,,,r,,,x,;,,d,e,u,x,,],[un,1 en lettres;de
```

En option, il est possible de préciser les couleurs de fond `color=[orange,black,white]` (défaut) : couleur des cases à remplir, couleur des cases "noires", couleur de fond du numéro du mot, de demander que les définitions soient en popup (option `tooltip`), que toutes les définitions s'affichent en même temps (option `allhelp`)

```
\answer{}{....}{type=crossword}{option=color=[orange,black,white] allhelp}
```

- **Mouvement dans un jeu d'échecs** (nom : `chessgame`) La première ligne de la bonne réponse est la position des pièces présentée (entre crochets) : `position des blancs ; position des noirs`. La seconde ligne de la bonne réponse est formée des mouvements de pièces demandés (entre crochets). La notation choisie est la notation algébrique internationale anglaise abrégée : `K` pour roi, `Q` pour reine, `R` pour tour, `B` pour fou, `N` pour cavalier et `P` pour pion. Aucune vérification de règles n'est faite. Il s'agit simplement de mouvements de pièces. Pour préciser la couleur (par défaut grise), mettre `color=nom_couleur`. dans le champ `option`.

```
\answer{}{[Kg1,g2,c3,f4,Rg8,Qh8;Qc7,f7,Rd6,c5,Kf5];[h8,e5;c3,c6]}{type=chessgame}{option=color=bro
```

Il peut y avoir plusieurs réponses possibles, elles sont séparées par le symbole `|`. Par exemple,

```
\answer{}{[Kg1,g2,c3,f4,Rg8,Qh8;Qc7,f7,Rd6,c5,Kf5];[h8,e5;c3,c6|c7,e5]}{type=chessgame}
```

signifie que la bonne réponse est soit deux déplacements successifs `h8,e5` puis `c3,c6` soit un déplacement `c7,e5`.

L'option `noanswer` permet de ne pas afficher l'échiquier correspondant à la bonne réponse dans la partie principale de l'exercice. Celle-ci s'affiche alors simplement sous forme codée dans l'analyse de la réponse (pour supprimer l'analyse de la réponse, on peut utiliser l'option générale `noanalyzeprint`).

La variable `reply` n contient sur chaque ligne (indiquée par un point-virgule) les items suivants

  - le mouvement effectué,
  - le nom des figures déplacées et (éventuellement remplacées) entre crochets,
  - le codage de l'échiquier après le mouvement.
- **Géométrie avec canvasdraw** (nom : `draw`).
Ce type est encore en développement (ne pas traduire encore la documentation).

Ce type de réponse permet à l'utilisateur de tracer des objets géométriques (parmi des points, cercles, droites, segments, flèches, rectangles, lignes brisées, polygons).

La bonne réponse doit être donnée sous la forme d'une matrice (le séparateur de lignes est le point-virgule `;'). La première ligne est le code dans la syntaxe canvasdraw/flydraw entre [ ]. La seconde ligne est la forme pouvant être dessinée parmi `points, circles, lines, segments, arrows, rects, polyline, polygon` suivie des bonnes réponses. Les coordonnées des points doivent être dans le repère mathématique défini dans le code de canvasdraw.

Les dimensions du dessin doivent être mises sur la première ligne du champ `\embed{}`.

Les options possibles sont pour l'instant la couleur (`color=_couleur_`) ainsi que les options de calcul de notes `eqweight split`.

Dans le cas où l'on désire analyser la réponse par des conditions, mettre le code `canvasdraw/flydraw` comme troisième ligne du champ de `embed`, la deuxième ligne étant le type de tracé par l'élève.

Exemples :

```
\text{canvas_code=xrange -4,4
  yrange -4,4
  opacity 155,155
  linewidth 1
  grid 1,1,grey
  opacity 255,255
  hline 0,0,black
  vline 0,0,black
  precision 10
  linewidth 2
  snaptogrid
}
\statement{
\embed{r1,200x200}
}
\answer{}{[\canvas_code];arrows,0,3,3,0,0,3,-3,0}{type=draw}{option=eqweight split}
```

```
\text{canvas_code=xrange -4,4
  yrange -4,4
  linewidth 1
  opacity 155,155
  grid 1,1,grey
  opacity 255,255
  hline 0,0,black
  vline 0,0,black
  precision 10
  linewidth 2
  snaptogrid
}

\statement{
  \embed{r1,200x200}
}
\answer{}{[\canvas_code];lines,0,0,3,0,0,0,3,3}{type=draw}{option=color=orange}
```

Exemple sans analyse :

```
1  \text{canvas_code=xrange -4,4
2    yrange -4,4
3    opacity 155,155
4    linewidth 1
5    grid 1,1,grey
6    opacity 255,255
7    hline 0,0,black
8    vline 0,0,black
9    precision 10
10   linewidth 2
11   snaptogrid
12 }
13
14 \statement{
15 \embed{r1,200x200
16 arrows
17 [\canvas_code]}
18 }
19 \answer{}{\rep}{type=draw}{option=eqweight split}
20
```

Éditeur actif

- **Géométrie dynamique** (nom : `geogebra`) Une applet de géométrie dynamique GeoGebra est affichée. L'utilisateur peut construire un certain nombre d'objets (points, droites, cercles, etc.). Ce type doit obligatoirement être utilisé avec la commande `\embed`

La bonne réponse doit être donnée sous la forme d'une matrice. Chaque ligne contient des instructions d'une des formes suivantes (f pour formal, n pour numeric)

```
n, "condition numérique", text
f, "condition formelle", text
n, "condition numérique avec syntaxe formelle", text
```

`text` apparaît dans l'analyse de réponse. Si aucun texte n'est mis, un texte par défaut est construit à partir de la condition ; si le texte est le mot `hiden`, la condition n'apparaît pas dans l'analyse de la réponse.

- **condition formelle** : permet de vérifier si des objets ggb ont été construits. doit être rédigée en anglais et dans la syntaxe de GeoGebra 5.0.182.0. (voir https://www.geogebra.org/manual/en/Geometry_Commands) Si le nom d'un objet ggb est imposé et que l'utilisateur répond avec un autre nom alors l'objet sera reconnu quand même mais une pénalité sera donnée. Il est cependant recommandé de mettre le nom dans le cas de nombreuses conditions. L'analyse de la réponse vérifie que la condition est bien réalisée sans remplacement. Quelques tests supplémentaires sont ajoutés. Par exemple, si la condition est `A=Center[c]`, on vérifiera aussi si l'utilisateur a répondu quelque chose comme `c=Circle[A,B]`.

  Exemples :

  - `Point` : les conditions formelles analysent la construction des nouveaux points. Par contre si un point est déjà construit sur la figure, les élèves doivent impérativement déplacer ce point pour que l'analyse s'effectue correctement.

```
\answer{}{f,A=Point,Le point A est construit :;
f,C=Point,Le point C est construit :;
f,F=Point,Le point F est construit :;}{type=geogebra}{option=max=6 extra=yes output=coord
```

Attention : il reste un problème de reconstruction quand un point est sur un des axes du repère mais l'analyse de la réponse reste correcte.

- `Circle` : les conditions formelles analysent l'ensemble des cercles qu'ils soient construits au préalable ou qu'ils soient construits par les élèves. Attention ce comportant différe des conditions sur un point vu au préalable. Pour éviter de transformer les cercles en ellipse, La commande setCoordSystem ne doit pas être utilisée ou judicieusement. Les conditions d'analyse sont les suivantes :
  - `c=Circle[A,B]` : vérifie que le cercle (c) de centre A et de rayon AB soit tracé. L'utilisateur doit impérativement avoir cliqué sur les points A et B avec l'outil cercle.
  - `=Circle[A,B]` : vérifie qu'un cercle quelconque de centre A et de rayon AB soit tracé. L'utilisateur doit impérativement avoir cliqué sur les points A et B.
  - `=Circle[,B]` : vérifie qu'un cercle quelconque passe par le point B qui n'est pas le centre du cercle. L'utilisateur doit impérativement avoir cliqué sur le point B.
  - `=Circle[C,]` : vérifie qu'un cercle quelconque a pour centre le point C. L'utilisateur doit impérativement avoir cliqué sur le point C.

```
\answer{}{f,O=Point,Le point O est construit :;
f,c=Circle[O,A],Le cercle \(C) de centre O et passant par A est construit :;
f,=Circle[,C],Un cercle passant par C (clic obligatoire sur C) est construit :;
f,=Circle[A,C],Un cercle de centre A et passant par C est construit :;
f,=Circle[C,],Un cercle de centre C (clic obligatoire sur C) est construit :;
}{type=geogebra}{option=output=nocoord extra=yes max=10}
```

- `Droite` : les conditions formelles prennnent en compte l'ensemble des droites construites au préalable ou réalisées par les élèves. Les conditions d'analyse sont les suivantes :
  - `a=Line[A,B]` : vérifie que la droite (a) passant par A et B soit tracé. L'utilisateur doit impérativement avoir cliqué sur les points A et B.
  - `=Line[A,B]` : vérifie qu'une droite passant par A et B soit tracé. Le nom de la droite ayant aucune importance permet de ne pas tenir compte de l'ordre de construction. L'utilisateur doit impérativement avoir cliqué sur les points A et B.
  - `=Line[,B]` : vérifie qu'une droite passant par B soit tracé. L'utilisateur doit impérativement avoir cliqué sur le point B avec l'object droite sélectionné.

```
\answer{}{f,B=Point,Le point B est construit :;
f,C=Point,Le point C est construit :;
f,a=Line[A,B],La droite (a) passant par les point A et B est construite :;
f,=Line[B,C],La droite (BC) est construite :;
f,=Line[,D],Une droite passant par le point D est construite :;
}{type=geogebra}{option=output=nocoord extra=yes max=10}
```

- `Perpendiculaire` : Les conditions formelles analysent l'ensemble des perpendiculaires. Pour conserver les angles, La commande setCoordSystem est à proscrire. De plus il est conseillé de mettre enableShiftDragZoom=false dans les options pour que les élèves ne modifient pas les échelles des repères. Les conditions d'analyse sont les suivantes :
  - `b=OrthogonalLine[A,a]` : vérifie que la droite (b) est perpendiculaire à (a) et passe par A. L'utilisateur doit impérativement avoir cliqué sur la droite (a) et le point A avec l'outil perpendiculaire.
  - `=OrthogonalLine[A,]` : vérifie qu'une droite perpendiculaire passe par le point A. L'utilisateur doit impérativement cliquer sur le point A avec l'outil perpendiculaire.
  - `=OrthogonalLine[,a]` : vérifie qu'une droite est perpendiculaire à la droite (a). L'utilisateur doit impérativement cliquer sur la droite (a) avec l'outil perpendiculaire.

L'ensemble des commandes disponibles sont les suivantes :

- `A=Point` : A est un point de la figure.
- `c=Circle[A,B]` : (c) est un cercle de centre A et passant par B.
- `a=Line[A,B]` : (a) doit être la droite passant par A et B.
- `a=LineBisector[A,B]` : (a) est la médiatrice du segment [AB].
- `b=PerpendicularLine[A,a]` : (b) est la perpendiculaire à (a) passant par A.
- `C=Middle[A,B]` : C est le milieu du segment [AB].
- `c=AngularBisector[a,b]` : (c) est la bisectrice des droites (a) et (b)
- `-30=AngleOriente[A,B,A,C]` : l'angle orienté (AB,AC) doit mesurer -30°.
- `30=AngleGeom[A,B,C]` : l'angle géométrique ABC doit mesurer 30°.

- **condition numérique** : les conditions numériques analysent les objets ggb à partir des coordonnées des points, des vecteurs, des équations de droites, des cercles... Dans le cas d'égalité, utiliser le test `==` . Il est recommandé de faire des tests d'inégalité numérique plutôt que des tests d'égalité : `abs(x_c-1.2)<10^(-6)` par exemple, plutôt que `x_c==1.2` .
  - `Point` : L'analyse numérique évalue le nom et les coordonnées des points. Les variables `x_A et y_A` sont respectivement l'abscisse et l'ordonnée d'un point A ou d'un vecteur A.
    La condition d'analyse est la suivante :
    `abs(x_A-1)<0.3&abs(y_A-3)<0.3 :` Vérifie le placement du point A avec une erreur absolue de 0.3 sur les valeurs de l'abscisse et de l'ordonnée.

```
\answer{}{n,abs(x_A-1)<0.3&abs(y_A-3)<0.3,Est-ce-que le point \(A(1,3)) est bien placé ?;
n,abs(x_C+2)<0.3&abs(y_C-5)<0.3,Est-ce-que le point \(C(-2,5)) est bien placé ?;
n,abs(x_F+4)<0.3&abs(y_F+3)<0.3,Est-ce-que le point \(F(-4,-3)) est bien placé ?;}{type=g
```

  - `Cercle` : l'analyse numérique d'un cercle nécessite selon les cas la connaissance du nom du cercle. Les variables `R_c, x_c et y_c` sont respectivement le carré du rayon d'un cercle c, l'abscisses et l'ordonné de son centre. Pour éviter de transformer les cercles en ellipse , La commande setCoordSystem ne doit pas être utilisée ou judicieusement.
    - Si la construction doit tolérer des approximations, les conditions d'analyse sont les suivantes :
      - `abs(R_c-((x_A-x_B)^2+(y_A-y_B)^2))<0.3 :` vérifie que le rayon du cercle (c) est environ égale à la distance AB. Attention R_c est le rayon au carré.
      - `abs(x_c-x_A)<0.1 & abs(y_c-y_A)<0.1 :` vérifie que le centre du cercle (c) est confondu avec le point A.
    - Dans le cadre d'une construction formelle, on peut réaliser l'analyse d'un cercle dont on ne connait pas le nom avec les conditions d'analyse suivantes :
      - `eq_c:(x+1)^2+(y-4)^2=10 :` vérifie que le cercle (c) a pour équation (x+1)^2+(y-4)^2=10
      - `eq_:(x-x_A)^2+(y-y_A)^2=(x_A-x_B)^2+(y_A-y_B)^2 :` vérifie qu'un cercle passe par le point A et a pour rayon AB.

```
\answer{}{n,abs(R_c-((x_A-x_B)^2+(y_A-y_B)^2))<0.3,Le rayon du cercle \(C) est correct :;
```

```
n,abs(x_c-x_A)<0.1 & abs(y_c-y_A)<0.1 ,Le centre du cercle \(C) est confondu avec le poin
n,eq_c:(x-x_A)^2+(y-y_A)^2=(x_A-x_B)^2+(y_A-y_B)^2, L'équation du cercle \(C) est bonne :
n,eq_:(x+1)^2+(y-4)^2=10, Le cercle de centre C est de rayon CA est tracé : ;}
{type=geogebra}{option=extra=yes max=20 output=nocoord}
```

- `Droite` : l'analyse numérique de droite nécessite selon les cas la connaissance du nom de la droite. Les variables `m_d et p_d` sont respectivement le coefficient directeur et l'ordonnée à l'origine de l'équation réduite de la droite (d). Les variables `a_d, b_d et c_d` sont les coefficients d'une équation de la droite (d) du type ax + by = c.
  - Si la construction doit tolérer des approximations, les conditions d'analyse sont les suivantes :
    - `abs(m_a-0.5)<0.1 & abs(p_a-2)<0.1 :` vérifie le coefficient directeur et l'ordonnée à l'origine de l'équation réduite de la droite (a) avec une précision de 0.1.
    - `abs(a_b-2)<0.1 & abs(b_b-1)<0.1 & abs(c_b+3)<0.1 :` vérifie les 3 coefficients de l'équation \(ax+by=c) d'une droite (b) avec une précision de 0.1. Attention une même droite peut avoir plusieurs triplets (a,b,c) donc il est préférable d'analyser les quotients.
  - Dans le cadre d'une construction formelle, on peut réaliser l'analyse d'une droite dont on ne connait pas le nom avec les conditions d'analyse suivantes :
    - `eq_:-2x+3y=-1 :` vérifie que la droite d'équation -2x+3y=-1 est construite. Par contre les coefficients de son équation ne sont pas analysé avec une erreur absolue. Par conséquent il est conseillé de créer un fichier ggb avec l'aimantation mise à "attaché à la grille" avec une grille relativement fine pour laisser de nombreuses possibilités.

```
\answer{}{
f,C=Point,Le point C est construit :;
f,=Line[A,B],La droite (AB) est construite :;
n,abs(m_a-0.5)<0.1 & abs(p_a-2)<0.1,Est-ce-que la droite (a) est bien tracée ?;
f,=Line[B,C],La droite (BC) est construite :;
n,abs(a_b-2)<0.1 & abs(b_b-1)<0.1 & abs(c_b+3)<0.1,Est-ce-que la droite (b) est bien trac
f,=Line[C,D],La droite (CD) est construite :;
n,eq_:-2x+3y=-1,Est-ce-qu'une droite d'équation \(-2x+3y=-1) est tracée ?;}{type=geogebra
```

- `Objet` : l'analyse numérique d'objet nécessite la connaissance de son nom. Les objets peuvent être du type `segment, function, angle, curseur et cellule du tableur`.

```
\answer{}{n,abs(v_a-3)<0.4,Est-ce-que le longueur du segment a est correcte ?}{type=geoge
\answer{}{n,v_f(x)=(1+x^2),Est-ce-que la fonction f est bien tracée ?}{type=geogebra}{opt
\answer{}{n,abs(v_a-\a)<0.1,Est-ce-que la curseur a est bien positionné ?}{type=geogebra}
\answer{}{n,abs(\test)<0.1 & abs(\test2)<0.1,Le vecteur \(\overrightarrow{u}\) est bien c
\answer{}{n,abs(v_B5-(\nwc))<=0,La case B5 est correct;
n,abs(v_D2-(\pm))<=0,Le prix Total HT des écrans est juste;}{type=geogebra}{option=extra=
```

- `x2_c,y2_c,xy_c,x_c,y_c,cst_c` : coefficients de `x^2, y^2, x*y, x,y, 1` d'une conique `c`.

On peut aussi tester des équations par exemple, `eq_a:2x+y=3` (ou `eq_:2x+y=3` si on ne désire pas donner d'importance au nom de la droite) teste si 2x+y=3 est une équation de la droite `a`.

Exemple : `x_c>0 && y_c>0 && R_c<4` demande de tester si le cercle `c` a un centre dont les coordonnées sont positives et un rayon inférieur à 2. L'utilisation des noms des objets pour formuler les *tests numériques* est obligatoire. Dans le cas d'égalité, utiliser le test `==`. Il est recommandé de faire des tests d'inégalité numérique plutôt que des tests d'égalité : `abs(x_c-1.2)<10^(-6)` par exemple, plutôt que `x_c==1.2`.

- **condition numérique avec syntaxe formelle** : La syntaxe est similaire à celle des conditions formelles mais l'analyse est faite numériquement avec une très grande précision. (par défaut 10^{-13}) Le logiciel pari est utilisé à l'aide des formules d'analyse classique des objets. Par exemple pour analyser la médiatrice du segment [AB], on utilise son équation : $2(x_B - x_A)x + 2(y_B - y_A)y + x_A^2 - x_B^2 + y_A^2 - y_B^2 = 0$. Par conséquent les noms des objets à l'intérieur des commandes doivent tous être donnés mais il n'est pas obligatoire de préciser le nom de l'objet construit. Le remplacement numérique est fait à l'aide des données numériques de la réponse de l'utilisateur.

```
\answer{}{n,LineBisector[A,B],La médiatrice du segment [AB] est correctement construite :;
}{type=geogebra}{option=extra=yes precision=2 max=150 output=noobjet}
```

Une seule réponse du type `geogebra` est possible pour l'instant dans l'exercice. On doit obligatoirement utiliser la commande `embed` :

```
\embed{reply1,option}
```

où `option` a comme première ligne la taille de l'applet en pixels (Largeur x Hauteur). Les lignes suivantes sont formées d'options de l'applet ou de commandes (objets prédéfinis). Par exemple :

```
showToolBar=true
customToolBar=0|40|2|10
A = (1,2)
B = (-1,2)
a = Line[A,B]
setFixed('A',true)
setFixed('B',true)
setAxesVisible(false,false)
```

La liste complète et à jour se trouve dans la documentation du slib geogebra.

```
file
ggbBase64
showToolBar (default : false)
showMenuBar (default : false)
showResetIcon (default : false)
showToolBarHelp (default : true)
```

```
enableRightClick (default : true)
enableLabelDrags (default : false)
enableShiftDragZoom (default : true)
showAlgebraInput (default : false)
customToolBar (Une barre d'outils est donnée par défaut. La liste des icônes se trouve là)
```

Les commandes possibles sont les définitions d'objets et certaines méthodes javascript de GeoGebra. Nous en donnons quelques-unes, voir la documentation du slib geogebra pour une liste à jour de ce qui est utilisable à travers WIMS.

```
evalCommand setValue setCoords setColor setVisible
setLabelVisible setLabelStyle setFixed setTrace
setAxesVisible setGridVisible setCoordSystem setUndoPoint
deleteObject renameObject setLayer setLayerVisible
setLineStyle setThickness setPointStyle setPointSize
setFilling setAnimating setAnimationSpeed
startAnimation stopAnimation isAnimationRunning
```

Par exemple,

```
evalCommand("SetFixed[Objet, true|false]")
```

Pour plus de détails, voir l'aide de GeoGebra sur la syntaxe des méthodes javascript disponibles :
http://www.geogebra.org/source/program/applet/geogebra_applet_javascript.html

La variable `\reply` *n* où *n* est le numéro du champ de réponse est formée de plusieurs lignes. La première ligne donne le nombre d'objets. Les lignes suivantes contiennent la définition de chaque objet (une ligne par objet), d'abord formellement si cela est possible, puis numériquement.

La variable `\sc_reply` *n* contient sur la première ligne une valeur. Elle évalue la réponse globale de la manière suivante :

- 0 : mauvaise réponse
- 0.5 : erreur de nom seulement
- 1 : bonne réponse

La seconde ligne est constituée de plusieurs colonnes qui évaluent chaque condition numérique ou formelle de la même manière que précédement. La troisième ligne indique le numéro du scénario utilisé. Si l'option n'est pas activé, le numéro renvoyé est 0.

Options :

- `max` nombre maximum d'objets qui peuvent être construits dans l'applet (par défaut : 10).
- `precision` : précision absolue (nombre de décimales prises en compte pour les comparaisons numériques, défaut : 18).
- `idontknow` (`yes` or `no` , par défaut : `no`) : fait apparaître un bouton `Je ne sais pas` à côté du bouton `Valider` .
- `extra` (`yes` or `no` , par défaut : `no`) : autorise les objets supplémentaires (si l'option `extra` n'est pas égale à `yes` , le score est diminué dans le cas où il y a plus d'*objets construits par l'élève* qu'il n'y avait de *conditions formelles* données dans la bonne réponse par l'auteur de l'exercice.)
- `output` : les valeurs possibles sont `coord` (défaut), `formal` , `nothing` . La différence se trouve dans la manière dont les objets construits par l'utilisateur sont réaffichés dans l'analyse de la réponse :
  - `coord` : les coordonnées des objets sont affichées ;
  - `nocoord` : seuls les noms des objets construits sont affichés ;
  - `formal` : les relations formelles sont affichées s'il y en a ;
  - `noobjet` : seul le texte écrit après les conditions est affiché.
  - `nothing` : aucun affichage.
- `weights` (par défaut, 1&0.2&1) le score final de la construction est la moyenne pondérée de trois scores dont on peut ici préciser les coefficients (séparés par des `&`). Ces trois scores sont respectivement le score des *tests formels* (f), le score des *noms des objets demandés*, le score des *tests numériques* (n).
- `object_analysis` : définit le préfixe des noms d'objets qui seront analysés. Il ne peut s'agir que d'objets prédéfinis. Dans ce cas, aucun autre objet ne sera analysé. Si un fichier ggb est utilisé, il est indispensable de modifier au préalable le fichier utilisé.
- `feedbackscript` : Sa valeur est le nom d'une fonction javascript qui doit être définie en javascript correct dans le champ `\feedback` . Lors de l'analyse de la réponse, un bouton permet de lancer son exécution. Par défaut le nom du bouton est Correction.
- `namebutton` : Sa valeur définit le nom du bouton.
- `scenario` : Cette option permet de définir plusieurs scenarios de constructions d'une figure. La première valeur est la ou les conditions nécessaires pour appliquer le scenario. Par conséquent on peut réaliser des scénarios qui dépendent des objets construits par l'élève. (à finaliser) Elle peut être vide et dans ce cas là le scénario est obligatoirement analysé. Les valeurs suivantes indiquent les lignes qui vont être analysées dans chaque scenario de construction. La note attribuée correspond toujours au scénario le plus favorable.

```
\answer{}{f,A=Point,Le point A est construit;
n,abs(x_A-1)<0.3&abs(y_A-3)<0.3,Est-ce-que le point \(A(1,3)) est bien placé ?;
f,B=Point,Le point B est construit;
f,C=Point,Le point C est construit;
f,D=Point,Le point D est construit;
f,E=Point,Le point E est construit;
f,F=Point,Le point F est construit;
n,=Line[A,B],la droite (AB) est construite;
f,=Line[B,C],la droite (BC) est construite;
}{type=geogebra}{option=extra=yes precision=3 max=200 output=noobjet feedbackscript=Correctior
scenario=@A:1:2:9@B:1:3:4:8@}
```

Remarques générales :

- L'utilisation du type de réponse `geogebra` ralentit le chargement des pages : il ne faut l'utiliser que si le recours à la géométrie dynamique est utile.
- Les objets dont le nom commence par `My_` ne seront pas reconnus par WIMS et ne seront pas réaffichés lors de l'analyse de la réponse.
- Les mesures des angles sont toujours en degrés.
- Une fonction `f` a pour nom `f(x)` . Il faut donc écrire quelque chose du type `f(x)=sin(x)` .

- **Dessiner une droite, un polygone, des points, un segment, un vecteur, un rectangle, une courbe.** (nom : `javacurve`)
Ce type de réponse permet à l'utilisateur de dessiner une droite, un polygone, un rectangle ou plusieurs points dans un dessin.

La bonne réponse doit être donnée sous la forme d'une matrice (le séparateur de lignes est le point-virgule `;'). La première ligne est l'URL de l'image. La deuxième ligne est formée d'une condition parmi la liste du tableau suivant. Si l'on désire analyser la réponse par des conditions, on mettra une variable vide pour la bonne réponse et on rajoute dans le champ `embed` en seconde ligne l'URl de l'image et en troisième ligne le type de tracé comme dans le tableau suivant (par exemple `points`, `line` ...).

Liste des figures disponibles

Syntaxe	Signification
`points,x1,y1,x2,y2,...`	Points en (x1,y1), (x2,y2), ... C'est un point ``épais", de largeur fixe.
`line,x1,y1,x2,y2`	Droite passant par (x1,y1), (x2,y2).
`sline,x1,y1,x2,y2`	Demi-droite passant par (x1,y1), (x2,y2), d'origine (x1,y1).
`polygon,x1,y1,x2,y2,...`	Polygone de sommets (x1,y1), (x2,y2), ...
`segment,x1,y1,x2,y2`	Segment d'extrémités (x1,y1), (x2,y2).
`vector,x1,y1,x2,y2`	Vecteur de (x1,y1) vers (x2,y2).
`rectangle,x1,y1,x2,y2`	Rectangle de diagonale (x1,y1), (x2,y2).
`rectangle,x1,y1,x2,y2,x3,y3,x4,y4`	Rectangle contenu dans la différence du rectangle de diagonale (x1,y1), (x2,y2) et du rectangle de diagonale (x3,y3), (x4,y4).
`circle,x1,y1,r`	Cercle de centre (x1,y1), rayon r.
`curve,x1,y1,x2,y2, ... ou curve, f(t), g(t) ou curve, f(x)`	Courbe (expérimental)

Des options peuvent être ajoutées dans le champ d'option : `precision`. Le paramètre `precision` doit être manié avec précaution ...

Il est recommandé d'insérer ce champ de réponse dans l'énoncé. La taille de l'image peut être indiqué en second argument de `\embed` sous la forme `L x H`, par exemple `\embed{reply` *n*`, 300 x 400}`.

En général, la variable `\reply` *n*, où *n* est le numéro du champ de réponse peut être utilisée dans un `\feedback` contient les variables correspondant à ce qui est rentré `(x1,y1 ...)` sauf dans les cas particuliers suivants :

- dans le cas de la condition `points`, la première ligne est la liste des coordonnées des points cliqués en pixels, la deuxième ligne est `n1,n2,n3` avec `n1` le nombre de points corrects, `n2` le nombre de points manquants, `n3` le nombre de points en trop. La troisième ligne est formée de la liste des numéros des points correctement trouvés.
- dans le cas de la condition `cercle`, la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2` avec `n1` égal à 1 si le centre est correct, à 0 sinon et `n2` égal à 1 si le rayon est correct, à 0 sinon.
- dans le cas de la condition `sline`, la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si l'origine est correcte, à 0 sinon, `n2` égal à 1 si la pente est correcte, à 0 sinon, `n3` égal à 1 si le sens est correct, à 0 sinon.
- dans le cas de la condition `sline`, la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si l'origine est correcte, à 0 sinon, `n2` égal à 1 si la pente est correcte, à 0 sinon, `n3` égal à 1 si le sens est correct, à 0 sinon.
- dans le cas de la condition `sline`, la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si la pente est correcte, à 0 sinon,

- **Sélectionner des atomes d'une molécule en 3D** (nom : `jmolclick`)

Une molécule est présentée dans l'applet **Jmol** et certains des atomes doivent être sélectionnés. Ce type de réponse doit être utilisé avec la commande `\embed{}`. D'autres objets 3D que des molécules peuvent être représentés grâce aux capacités de **Jmol**

La bonne réponse est formée de deux lignes (séparateur ;) : la première est la bonne réponse, la seconde indique la molécule à montrer : le fichier de structure de la molécule (dans un format reconnu par **Jmol**) peut être dans un répertoire `data` du module, dans le répertoire data de la distribution WIMS (adresse **`data/xxx`**) ou dans un module de données dans **`modules/data`** (dans ce cas, l'adresse est de la forme **`datamodule/xxx`**) ou dans le répertoire **`images`** (l'adresse est alors **`\imagedir/xxx`**). On peut aussi mettre simplement le texte décrivant la molécule. **Jmol** peut aussi interroger le serveur du *National Cancer Institute*. Ce serveur permet à **Jmol** de charger une molecule à partir de son nom (anglais) ou de son code SMILES. Pour cela, il suffit de faire précéder le nom ou le code SMILES d'un @.

La première partie de la bonne réponse est donnée par la liste des numéros des atomes à sélectionner (comme dans le fichier de structure) ou par une commande dans la syntaxe **Jmol** (des exemples sont donnés ci-dessous). Ainsi, `1, 2, 3` est équivalent à `atomno=1 or atomno=2 or atomno=3` ou encore à `({0,1,2})` (les deux dernières étant en syntaxe **Jmol**).

Il est possible de donner plusieurs réponses possibles ; dans ce cas-là, elles doivent être séparées par un `|||` (afin de ne pas confondre avec le `|` qui peut être utilisé dans la syntaxe **Jmol**). Attention : si `no_select=2`, `_0 ||| 1|2|4` signifie qu'on doit sélectionner deux atomes d'oxygène (exactement) ou deux des atomes parmi les numéros 1, 2 et 4.

La première ligne de la commande `\embed` est formée de la taille X x Y en pixels de l'applet. Sur les trois lignes suivantes, des scripts **Jmol** (cf la documentation de Jmol) peuvent être mis : le premier sera exécuté lors de la première requête de l'exercice ET lors de la réponse. Le deuxième sera exécuté lors de la première requête de l'exercice uniquement. Le troisième sera exécuté uniquement lors de la réponse. Il est aussi possible de donner l'adresse d'un fichier contenant un script. Les mêmes règles que pour le fichier de structure de la molécule sont applicables (**`data/scriptsimple.spt`**). Pour être reconnu, le fichier doit avoir l'extension `.spt`.

Exemple complet:

```
\title{Exemple simpliste (et idiot) de jmolclick}
\statement{
Sélectionnez les atomes d'oxygène.
\embed{r1, 300x300
  select atomno=3; spacefill 0.8;select none;
 set echo top left; echo "avant la réponse";
 set echo bottom right; echo "après la réponse";}
}
\answer{}{5,6;@cafeine}{type=jmolclick}
```

Exemples (partie embed uniquement):
Un exemple très simple:

```
\embed{r1, 200x200}
```

Exemple avec un script explicite (commun à la question et à la réponse) :

```
\embed{r1, 200x200
 select all; wireframe 0.2;select none;
 }
```

Exemple avec un script dans un fichier (dans le dossier `data` du module)

```
 \embed{r1, 300x300
 data/scriptsimple.spt
 set echo top left; echo "avant la réponse";
 set echo bottom right; echo "après la réponse";}
```

Il est aussi possible lorsque ce script est créé par l'exercice (par exemple, lorsqu'il contient des variables aléatoires) d'utiliser le slib `oef/newfile`
exemple :

```
\text{filescript=slib(oef/newfile spt,\text_script)}
.
.
.
\embed{r1, 200x200
```

où la variable `text_script` contient le texte du script. La variable `filescript` contient l'adresse du fichier créé.

Exemples (partie answer uniquement): Avec le nom courant

```
\answer{}{_0;@tnt}{type=jmolclick}
```

ou avec le nom IUPAC

```
\answer{}{_0;@2,4,6-Trinitrotoluene}{type=jmolclick}
```

ou encore avec le code SMILES

```
\answer{}{_0;@Cc1c(cc([N+](=O)[O-])cc1[N+](=O)[O-])[N+](=O)[O-]}{type=jmolclick}
```

ou enfin à l'aide d'un fichier .mol (qui serait dans le dossier data du module contenant l'exercice)

```
\answer{}{_0;data/trinitrotoluene.mol}{type=jmolclick}
```

Le dernier format est à privilegier car il est indépendant de l'accès à la base de données NCI.

Les options suivantes du type de réponse `jmolclick` sont possibles :

`split` ou `partialscore`	Le score tient compte d'une réponse bonne partielle, mais surpénalise les mauvaises réponses.
`eqweight`	les mauvaises réponses ont le même poids que les bonnes réponses.
`noselect="liste de numéros d'atomes"`	les atomes en question ne pourront pas être sélectionnés.
`nb_select=` m	Contraint à ce que le nombre d'atomes sélectionnés soit exactement `m`. Un message d'alerte invite à corriger dans le cas contraire.
`target=` numéro	Si plusieurs applets (soit jmolclick, soit jmolshow) se trouvent dans votre exercice, elles doivent toutes avoir des numéros différents et la première doit avoir le numéro 0. Ce numéro est utile pour associer des boutons (ou autre) à une applet. Le numéro par défaut est 0.
`type=Java`	Pour experts : La version Java de **Jmol** sera utilisée mais le chargement de structures liées à serveur externe (structures précédées de @) est impossible (la présence du symbole @, fait basculer automatiquement vers la version sans Java).
`applet=imagedir`	Pour experts : l'applet utilisée sera celle mise dans le répertoire `images` du module. N'utilisez cette solution que si vous avez réellement besoin d'une version plus récente de **Jmol**.

La variable `reply`n contient trois listes séparées par des points-virgule : la liste des bonnes réponses de l'étudiant; la liste des réponses fausses; la liste des atomes oubliés. Dans le cas où le premier champ de la bonne réponse est une variable de contenu vide, la variable `reply`n contient simplement la liste des atomes sélectionnés.

Exemples de réponses Jmol (première ligne du deuxième champ de `answer`)

Syntaxe	Explications
`atomno=1`	`l'atome numéro 1`
`@1`	`l'atome numéro 1`
`({1})`	`l'atome numéro 2 (attention au décalage de 1).`
`atomno=1 or atomno=2 or atomno=5`	`les atomes 1, 2 et 5`
`({0 1 4})`	`les atomes 1, 2 et 5 (attention au décalage de 1).`
`({0 1 4:7})`	`les atomes 1, 2 et les atomes de 5 à 8 (attention au décalage de 1).`
`_C`	`tous les atomes de carbone (on peut aussi mettre carbon ou utiliser le numéro atomique elemno=6 )`
`connected(2)`	`tous les atomes ayant deux liaisons (quelque soit le type de liaisons)`

connected(2,4)	tous les atomes ayant entre 2 et 4 liaisons (quelque soit le type de liaisons)
connected(double)	tous les atomes ayant une liaison double (on peut aussi mettre single , triple ou aromatic)
connected(atomno=5)	tous les atomes connectés à l'atome numéro 5
connected(2,nitrogen)	tous les atomes connectés exactement à deux atomes d'azote (ils peuvent être connectés à d'autres types d'atomes)
within(2.1,atomno=4)	tous les atomes à une distance de moins de 2.1 Å de l'atome numéro 4
within(2.1,plane,@{plane({atomno=3},{atomno=4},{atomno=5})})	tous les atomes à moins de 2.1 Å du plan défini par les atomes numéro 3, 4 et 5 (attention, il s'agit d'une mesure algébrique)
within(2.1,plane,@{plane({atomno=3},{atomno=4},{atomno=5})}) or within(-2.1,plane,@{plane({atomno=3},{atomno=4},{atomno=5})})	tous les atomes à moins de 2.1 Å du plan défini par les atomes numéro 3, 4 et 5.

Les conditions peuvent être combinées dans **Jmol** par les opérateurs d'ensembles `OR`, `AND` et `NOT`. Pour plus de détails, voir la documentation de Jmol.

Exemples de scripts Jmol

Syntaxe	Explications
`connect (({4}) ({5}) double modifyorcreate;`	Crée une liaison double entre les atomes numéro 4 et 5
`connect (({4}) ({5}) delete;`	Détruit les liaisons entre les atomes numéro 4 et 5
`connect (_C) delete;`	Détruit les liaisons entre les atomes de carbone
`select (_N); color atoms green; select none;`	Colorie en vert les atomes d'azote
`select ({5}); spacefill 0.1; select none;`	réduit la taille de l'atome numéro 5
`select all; wireframe 0.05; select none;`	réduit l'épaisseur des liaisons

Des slibs sur Jmol sont accessibles. slib

- **Géométrie avec jsxgraph** (nom : `jsxgraph`).
Une applet de géométrie dynamique jsxgraph est affichée. L'utilisateur peut déplacer des points. Les nouvelles coordonnées des points sont analysées. Ce type de réponse est technique car le développeur doit connaître et utiliser le langage javascript de jsxgraph afin de dessiner la figure.

  Il est obligatoire d'utiliser la commande `\embed` pour insérer le champ de réponse dans le texte :

  ```
  \embed{reply1,option}
  ```

  Description de `option` :
  - La première ligne est la taille de l'applet en pixels (Largeur x Hauteur).
  - La deuxième ligne est le mot permettant de définir le "board" de JSXGraph (dans l'exemple suivant, c'est `jxgbox`). Ce mot doit se retrouver dans le nom des variables dans ce qui suit.
  - La troisième ligne est le script permettant l'affichage de l'applet. Les points qui doivent être modifiés par l'utilisateur doivent être définis sous la forme

    ```
    jxgbox_rep1 = brd.create('point',jxgbox_var1 );
    ```

  - La quatrième ligne donne les positions initiales des variables `jxgbox_var1` (séparées par des points virgules).

  La bonne réponse est formée des positions `x,y` des points `jxgbox_var1`, ... séparées par des points virgules. Si un de ces points doit rester une réponse libre, il suffit de la remplacer par la valeur d'une variable non encore définie (dans l'exemple suivant, `\c`). Il est alors possible de compléter l'analyse par des conditions supplémentaires en utilisant la commande `\condition{}{}`.

  On peut aussi permettre de modifier la position d'un curseur. Dans ce cas, la réponse est simplement un nombre. Dans le cas d'une réponse libre, il est nécessaire de préciser qu'il ne s'agit pas d'un point mais d'un nombre en faisant suivre la variable du nombre de réels attendus (séparée par `:`), par exemple,

  ```
  \answer{}{\rep1:1;\rep2:2}{type=jsxgraph}
  ```

  Par défaut, le nombre est 2.

  Les options possibles sont pour l'instant la précision (absolue).

```
\title{Exemple de jsxgraph}
\precision{100}
\text{a=randint(1..5), randint(1..5)}
\text{b=randint(1..5), randint(1..5)}
\text{script= var brd = JXG.JSXGraph.initBoard('jxgbox', {
 axis:true,boundingbox: [-6, 6, 6, -6], grid:true});
jxgbox_rep1 = brd.create('point',jxgbox_var1 );
jxgbox_rep2 = brd.create('point',jxgbox_var2 );
jxgbox_rep3 = brd.create('point',jxgbox_var3 )}

\statement{Mettre le point A en (\a). Mettre le point B en (\b).
Mettre le point C sur la droite AB.
 \embed{r1,400x400
jxgbox brd
\script
jxgbox_var1=[0,0] ; jxgbox_var2=[1,0]; jxgbox_var3=[0,1]
  }
}

\answer{}{\a;\b;\c}{type=jsxgraph}{option=precision=10}

\matrix{rep=\reply1}
\real{test=abs((\rep[3;1]-\a[1])*(\b[2]-\a[2]) - (\rep[3;2]-\a[2])*(\b[1]-\a[1]))}
\condition{C est sur AB}{ \test < 0.1}
```

- **Dessiner une droite, un polygone, des points, un segment, un vecteur, un rectangle, une ligne brisée** (nom : `jsxgraphcurve`)
Ce type de réponse permet à l'utilisateur de dessiner une droite, un polygone, un rectangle ou plusieurs points dans un dessin.

La bonne réponse doit être donnée sous la forme d'une matrice (le séparateur de lignes est le point-virgule `;'). La première ligne est l'URL de l'image. La deuxième ligne est formée d'une condition parmi la liste du tableau suivant. Si l'on désire analyser la réponse par des conditions, on mettra une variable vide pour la bonne réponse et on rajoute dans le champ `embed` en seconde ligne l'URl de l'image et en troisième ligne le type de tracé comme dans le tableau suivant (par exemple `points` , `line` ...).

Liste des figures disponibles

Syntaxe	Signification
`points,x1,y1,x2,y2,...`	Points en (x1,y1), (x2,y2), ... C'est un point ``épais'', de largeur fixe.
`line,x1,y1,x2,y2`	Droite passant par (x1,y1), (x2,y2).
`sline,x1,y1,x2,y2`	Demi-droite passant par (x1,y1), (x2,y2), d'origine (x1,y1).
`polygon,x1,y1,x2,y2,...`	Polygone de sommets (x1,y1), (x2,y2), ...
`segment,x1,y1,x2,y2`	Segment d'extrémités (x1,y1), (x2,y2).
`vector,x1,y1,x2,y2`	Vecteur de (x1,y1) vers (x2,y2).
`rectangle,x1,y1,x2,y2`	Rectangle de diagonale (x1,y1), (x2,y2).
`rectangle,x1,y1,x2,y2,x3,y3,x4,y4`	Rectangle contenu dans la différence du rectangle de diagonale (x1,y1), (x2,y2) et du rectangle de diagonale (x3,y3), (x4,y4).
`circle,x1,y1,r`	Cercle de centre (x1,y1), rayon r.
`curve,x1,y1,x2,y2, ... ou curve, f(t), g(t) ou curve, f(x)`	Courbe (expérimental)
`brokenline,x1,y1,x2,y2, ...`	Ligne polygonale

Des options peuvent être ajoutées dans le champ d'option : `precision` , `color1` (couleur de la réponse donnée), `color2` (couleur de la bonne réponse). Par exemple

```
\answer{}{...}{type=jsxgraphcurve}{option=color1=black color2=#B4B4FF precision=8}
```

Le paramètre `precision` doit être manié avec précaution ...

Il est recommandé d'insérer ce champ de réponse dans l'énoncé. La taille de l'image peut être indiqué en second argument de `\embed` sous la forme `L x H` , par exemple `\embed{reply` *n*, `300 x 400}` .

En général, la variable `\reply` *n*, où *n* est le numéro du champ de réponse peut être utilisée dans un `\feedback` contient les variables correspondant à ce qui est rentré `(x1,y1 ...)` sauf dans les cas particuliers suivants :

- dans le cas de la condition `points` , la première ligne est la liste des coordonnées des points cliqués en pixels, la deuxième ligne est `n1,n2,n3` avec `n1` le nombre de points corrects, `n2` le nombre de points manquants, `n3` le nombre de points en trop. La troisième ligne est formée de la liste des numéros des points correctement trouvés.
- dans le cas de la condition `cercle` , la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2` avec `n1` égal à 1 si le centre est correct, à 0 sinon et `n2` égal à 1 si le rayon est correct, à 0 sinon.
- dans le cas de la condition `sline` , la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si l'origine est correcte, à 0 sinon, `n2` égal à 1 si la pente est correcte, à 0 sinon, `n3` égal à 1 si le sens est correct, à 0 sinon.
- dans le cas de la condition `sline` , la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si l'origine est correcte, à 0 sinon, `n2` égal à 1 si la pente est correcte, à 0 sinon, `n3` égal à 1 si le sens est correct, à 0 sinon.
- dans le cas de la condition `sline` , la première ligne est la liste des coordonnées des points cliqués en pixels et du rayon, la deuxième ligne est `n1,n2,n3` avec `n1` égal à 1 si la pente est correcte, à 0 sinon,

Si l'on désire centrer le résultat, on doit définir les classes css jxgbox et jsxgraph_button. Par exemple,

```
<style type="text/css">
.jxgbox {margin-left:auto;margin-right:auto}
.jsxgraph_button {text-align:center;}
</style>
```

Exemple :

```
\title{Exemple de jsxgraphcurve}

\title{ Droite}
\precision{100}
\css{<style type="text/css">
.jxgbox {margin-left:auto;margin-right:auto;}
.jsxgraph_button {text-align:center;}
</style>}

\text{dessin=draw(200,200
hline 100,100,black
vline 100,100,black
hline 100,50,black
vline 150,100,black
text black,100,100,large,O
text black,150,50,large,A
)
}
\statement{ Tracer la droite passant par 0 et par A.
 \embed{r1,200x200}
}
\answer{}{\dessin;line,100,100,150,50}{type=jsxgraphcurve}
```

- **Texte (sensible à la casse).** (nom : `keyboard`) permet d'utiliser un clavier, la réponse doit être exactement la bonne réponse.

  Un clavier est proposé. Les claviers préparés contiennent les lettres non ascii des languages correspondants : allemand (de), anglais (en), espagnol (es), français (fr), grec (el), italien (it), polonais (pl), russe (ru), slovène (si), phonétique IPA anglais (en_ipa), phonétique IPA français (fr_ipa). Ils sont accessibles par l'option `keyboard=it`. On peut aussi proposer ses lettres en mettant le code html en option sous la forme

  ```
  option=keyboard=[&#913; &#914; &#915; &#916; &#917;;
  &#918; &#919; &#952; &#921; &#922;;
   &#945; &#946; &#947; &#948; &#949; &#950; &#951; &#920; &#953; &#954;]
  ```

  Vous pouvez aussi utiliser un clavier universel (option `keyboard=universal` ou `keyboard=universal_xx` avec yy le clavier de la langue désirée, par exemple `ar`, `ru`, `ja`, `zh`).

  Si la réponse utilise des caractères non iso-latin1, vous devez les entrer par le code html pour l'instant. Cependant, un copier-coller de ces caractères dans la fenêtre de Createxo les transformera automatiquement en code html. La réponse est comparée comme texte à une bonne réponse donnée. Chaque mot de la réponse doit être exactement le même que le mot correspondant de la bonne réponse.
- **Matrice.** (nom: `matrix`)
  Comme pour les réponses vectorielles, chaque coefficient est évalué et comparé avec le coefficient correspondant de la bonne réponse.

  Trois mots d'options sont possibles : `split_coeff`, `split_row`, `split_column`. La réponse est alors considérée comme partiellement juste selon que un pourcentage suffisant de bons coefficients, ou de bonnes lignes ou de bonnes colonnes (50 %) est correct.

  Si `n` est le numéro de la question, la variable `result`n contient une matrice de 1 ou de 0 indiquant la position des bons et des mauvais coefficients (ou 0.5 si la précision n'est pas suffisante).

- **Nombre rationnel.** (nom : `numexp`)
  Comparaison de nombres rationnels. L'écriture fractionnaire doit être irréductible sauf si l'option `noreduction` est précisée.
  L'écriture décimale ou fractionnaire est acceptée pour les nombres décimaux. (0.25 pour 1/4 mais pas 0.33333333333333 pour 1/3).
- **Reconstituer une image présentée comme un puzzle.** (nom: `puzzle`)

  La bonne réponse contient deux champs, séparés par un point-virgule `;`. En général, le premier champ est le nom de l'image (qui doit être dans le dossier indiqué par la variable `common_images`), le deuxième champ `N1 x N2 x W` indique que l'image sera divisée en N1 lignes et N2 colonnes et que la largeur des morceaux du puzzle sera de `W`. La taille totale de l'image (largeur x hauteur) doit être fournie dans le premier champ de `\embed` : `\embed{r1, L x H}`.

  Il est aussi possible de fournir une image déjà découpée. Dans ce cas, le deuxième champ est comme précédemment `N1 x N2 x W`, mais le premier champ est la liste des adresses des images dans l'ordre suivant : première ligne de gauche à droite, deuxième ligne de gauche à droite, etc. (N1*N2 images doivent donc être fournies). Si le champ `option` contient le mot `text`, il est possible dans ce cas de remplacer toutes les images par du texte (sans virgule, ni point-virgule).

  La variable `reply` i contient une suite de 1 et de 0 selon que le morceau correspondant est bien placé ou non.

  Attention que deux morceaux ne soient pas identiques lors du découpage (par exemple, vides).
- **Nombre avec zone.** (nom: `range`)
  Comparaison de nombres réels dans une zone : la bonne réponse est composée de deux nombres, m et n, séparés par une virgule, avec m<n. Toute entrée comprise entre m et n (inclusive) sera acceptée.

  Des intervalles multiples peuvent être donnés en ajoutant plus d'items dans la bonne réponse.

  Si la bonne réponse contient un nombre impair d'items, le dernier sera utilisé pour montrer la bonne réponse.

  La macro slib `text/approximation` peut être utilisée pour calculer ces trois nombres automatiquement.
- **Mise en ordre d'objets donnés.** (nom : `reorder`)
  Liste des objets donnés selon un "bon" ordre.
  Les objets peuvent être du texte (mots ou groupes de mots), des formules (formatées via \(...)), ou des images. La bonne réponse donnée doit lister les objets comme items, dans l'ordre correct. Sinon, le séparateur est la virgule. Le serveur présentera les items dans un ordre aléatoire. Si la bonne réponse possède une deuxième ligne, celle-ci servira de séparateur entre les objets.

  Ce type, fondé sur DynAPI, peut ne pas fonctionner avec certains navigateurs.

- **Ensemble fini (textuel, formel et approximatif).** (names: `set`, `fset` et `aset`) C'est une liste d'éléments séparés par des virgules dont l'ordre n'a pas d'importance.

  Pour `set`, aucune évaluation n'est effectuée avant la comparaison (donc 2 et 1+1 ne sont pas pareils). Si le mot `repeat` est présent dans l'option, les répétitions des éléments sont demandées comme dans la bonne réponse. Sinon, la réponse est considérée comme partiellement bonne.

  Pour `fset` les éléments sont évalués en tant qu'*expressions formelles*.

  Pour `aset`, les éléments sont évalués en tant qu'*expressions approximatives*.

  Il est possible de présenter un champ pour chaque objet de l'ensemble en mettant le mot `distinct_inputs` dans l'option. Dans ce cas l'option `repeat` est automatiquement activée.

- **Données physiques avec nombre précis de chiffres significatifs.** (nom : `sigunits`)
  La réponse de type texte est comparée à une bonne réponse donnée. La syntaxe est <nombre> [<unité physique>] [#<nombre prescrit de chiffres significatifs>] [,<unité physique souhaité à l'affichage>]

  Exemple:
  - **1m#3** signifie 1 mètre avec 3 chiffres significatifs. Synonymes: *1.00m*, *0.00100 km*.
  - **100.0 N** a la même signification que *1e2N#3*: 100 newtons avec 3 chiffres significatifs.
  - **0.50A.h** donnerait "1.8e+02 C" à l'affichage. Utiliser **0.50A.h** pour avoir "5.0e-01 A.h".
- **Symtext.** (nom: `symtext`)

  Ce type est activé soit par la définition du type soit par la déclaration de l'option `symtext` dans d'autres types de réponse. Dans le premier cas, le champ de réponse est un `textarea` permettant d'entrer plusieurs lignes de texte. Dans le deuxième cas, le champ de réponse reste le même que pour le type de départ.

  La bonne réponse est une phrase sous syntaxe symtext (on pourrait souvent se contenter du style generic). Exemples (style `generic`) :

  - `a _ou b` permet de reconnaitre "a ou b", "b ou a", "soit a soit b", "soit b soit a", "a ou bien b", "b ou bien a".
  - `André _et [mon prof]` permet de reconnaitre "André et mon prof", "mon prof et André", "André ainsi que mon prof", "mon prof ainsi qu'André".
  - `_je _etre _plus grand que [mon pere]` permet de reconnaitre "je suis plus grand que mon père", "je suis moins petite que mon père", "mon père est plus petit que moi", "mon père est moins grand que moi".
  - `x _egale y` permet de reconnaitre "x égale y", "y est égal à x", "x et y sont égaux", "y et x sont en égalité", etc.

  Il est recommandé de mettre un premier alternatif en texte pur. Par exemple, si la bonne réponse est "A et B" sans tenir compte de l'ordre, il vaux mieux mettre la bonne réponse en `A et B|A _et B` au lieu de `A _et B` seulement. Dans ce cas, c'est le premier alternatif qui sera affiché comme bonne réponse aux élèves, au lieu de la syntaxe symtext elle-même, cette dernière pouvant troubler les élèves.

  Si une seule ligne est présente dans la déclaration de la bonne réponse, tout ce qui ne s'identifie pas à cette ligne sera considéré comme mauvaise réponse. Par contre si la bonne réponse donnée contient plusieurs lignes (séparées par le point-virgule '`;`'), toute réponse qui s'identifie à une ligne ultérieure sera considérée comme fausse, et toute réponse contenant un mot qui n'apparait dans aucune des lignes sera considérée comme incompréhensible et renvoyée pour retaper. Enfin, une réponse contenant que des mots compréhensibles mais qui ne s'identifie par aucune des lignes sera fausse.

  Normalement, il suffit de mettre la liste de mots qu'on veut reconnaitre dans la ligne 2, comme pour les types `case`, `nocase` ou `atext`.

  Il y a aussi la possibilité de déclarer des réponses partiellement correctes. Il suffit pour cela de déclarer ces réponses partiellement correctes à partir de la ligne 2, puis déclarer `goodlim=n` dans l'option, où `n` doit être un entier plus grand que 1. Dans ce cas, une réponse qui s'identifie à une ligne entre 2 et `n` sera partiellement correcte, avec une notation qui diminue avec le nombre de la ligne.

  Toutes les options de symtext sont reconnues dans le champ d'option.

  La sortie du programme symtext est disponible par la variable `\resultn`, où *n* est le numéro du champ de réponse, et peut être utilisée dans un `\feedback`.

  Détail de la syntaxe symtext.

- **Nombre avec unité.** (nom: `units`)
  Typiquement c'est une quantité physique avec une unité attachée. La réponse sera évaluée en tant que telle, telle que par exemple `1.5 cm` et `15 mm` signifient la même chose. L'exigence de précision est la même que pour l'expression numérique.
- **Vecteur.** (nom: `vector`)
  Chaque coefficient est évalué et comparé avec le coefficient correspondant de la bonne réponse. Seuls les vecteurs numériques peuvent être utilisés ici. La réponse et la bonne réponse peuvent avoir ou ne pas avoir une paire de parenthèses qui les entourent.
- **Mots d'une liste.** (nom : `wlist`)
  Dans ce cas, la réponse de l'utilisateur est comparée mot pour mot avec la bonne réponse, sans tenir compte de l'ordre. Et la comparaison n'est faite que sur les éléments essentiels des mots : les différences majuscule/minuscule, certaines différences singulier/pluriel (s en fin de mot), les accents sur les lettres, les mots très communs en anglais (a, the, ...) sont ignorés.

  Si le premier mot de la bonne réponse donnée est un nombre positif n, cela signifie que la réponse de l'utilisateur doit avoir au moins n mots différents égaux à ceux listés dans la bonne réponse donnée.

  Pour que la réponse de l'élève soit obligatoirement contenue dans un champ lexical donné, on peut définir des mauvaises réponses en utilisant le point virgule `;` comme séparateur puis le tube `|` pour mettre plusieurs mauvaises réponses. Si la réponse donnée par l'élève n'est pas incluse dans ce champ lexical, il est demandé à l'élève de compléter à nouveau la réponse pour forcer l'utilisation d'un des mots du champ lexical. Les signes de ponctuation seront ignorés dans cette liste et les mots sont comparés avec le même niveau de tolérance que pour les bonnes réponses.

- **Mémoire** (nom : `time`)
  Le type de réponse `time` demande simplement de regarder l'exercice pendant un certain temps. Il s'agit donc d'un préliminaire à une question faisant appel à ce qui a été vu lors de cette première étape. Les paramètres d'entrée sont trois nombres représentant des temps en millisecondes: le minimum de temps de lecture, le maximum de temps de lecture, le maximum de temps de lecture toléré. Si le temps est inférieur au minimum ou supérieur au maximum toléré, le score sera de 0. Dans le cas où le temps est compris entre le temps minimum et le temps maximum, le score est de 10, sinon la réponse est considérée comme partiellement bonne.

```
\answer{}{1000,2000,15000}{type=time}
```

```
\text{word=randitem(jouet,jouer,joue)}
\steps{r1
r2}
\statement{
```

```
\if{\step=1}{ Regardez le mot suivant : \word \embed{r1} }
\if{\step>=2}{ Ecrire le mot que vous avez lu :
\embed{r2} }
}
\answer{}{1000,2000,15000}{type=time}
\answer{}{\word}{type=text}
```

Pour limiter le temps de réponse d'un exercice, mieux vaut utiliser le chronomètre lors de la fabrication de la feuille.

Insertion des champs de réponse dans le texte de l'énoncé

Il est possible d'inclure des champs de réponse dans l'énoncé des exercices sous format html par la commande `embed`. Les réponses et choix doivent être définis de façon habituelle. Des conditions particulières sont indiquées dans la documentation des types de réponses.

- `\embed{choice 1}` insérer le premier *choix multiple* dans le texte.
- `\embed{reply 2}` insérer la deuxième *réponse libre* dans le texte.
- `\embed{reply 1,5}` peut avoir plusieurs significations suivant le type de réponse. En général, cela signifie : insérer la première *réponse libre* avec la taille du champ de réponse égale à 5. Par contre, si le type de `reply 1` est parmi `checkbox`, `click`, `radio`, `\embed{reply 1,5}` insère seulement l'un des choix (le numéro 5 dans ce cas) dans le format correspondant au type. Cela permet l'insertion de choix dans différentes structures de texte (en liste, dans un tableau, ...).
- `\embed{reply 1,AxBxC}` peut avoir plusieurs significations selon le type :
  - dans le cas où le type est `clickfill` ou `dragfill`, A et B sont les tailles horizontale et verticale (en pixels) d'une case, et C est le nombre de cases que contient le champ de réponses.
  - dans le cas où le type est `correspond`, A est la taille verticale des items, B et C sont respectivement la taille horizontale des colonnes de gauche et de droite.
  - Dans le cas de réponses de type texte libre, d'autres lignes peuvent être ajoutées permettant de personnaliser le style du champ de réponses. De plus, si la première ligne supplémentaire est le mot `default`, le champ aura la classe de style css `wims_oef_input` : le gestionnaire du site ou l'enseignant peuvent alors le configurer de manière générale. Par exemple, pour le type `numeric`,

    ```
    \embed{reply 1,12
    default}
    ```

    ou

    ```
    \embed{reply 1,12
    style="font-size:18px;background-color:#e8ffff;border:1px solid #3333CC;"}
    ```

- `\embed{reply 1,\i, \pick[\i;]}` Dans le cas où le type est un choix multiple, insère le choix n° `\i` dans le texte, mais en lui donnant l'apparence "`\pick[\i;]`". Typiquement, cela permet d'avoir des choix assez simples (A,B,C), mais dont l'apparence est complexe (formules mathématique par exemple).

Le paramètre optionnel `weight`

Il permet d'imposer un poids aux questions posées.

Exemple :

```
\answer{}{1}{type=numeric}{weight=8}
\answer{}{1}{type=numeric}{weight=2}
```

Le niveau de sévérité sera aussi pris en compte lors du calcul fait et le pourcentage sera pris sur un pourcentage fixé par le niveau de sévérité.

Il est possible de mettre des variables dans `weight` dans la mesure où elles sont définies avant le `\statement`.

Exemple :

```
\text{w = \t = 1 ? 1 : 2}
\answer{}{1}{type=numeric}{weight=\w}
\answer{}{1}{type=numeric}{weight=3}
```

Le paramètre optionnel `option`

Dans le cas de plusieurs options, elles sont séparées par des espaces. Par exemple, `option=default noanalyzeprint` Les options possibles à la suite d'une instruction `\answer` ou `\condition` sont

- `option=default` : L'option `default` est commune à toutes les réponses libres : cette option peut prendre la forme `default="valeur_par_défaut"`. Si l'utilisateur donne une réponse vide, c'est la valeur_par_défaut qui remplace.
- `option=split` : Dans le cas d'une réponse dont le type est `correspond`, `checkbox` ou `mark`, la notation tient compte des réponses justes, même si elles ne le sont pas toutes.
- `option=nonstop` : Dans le cas d'un exercice à étapes, les questions suivantes sont posées même en cas de réponses fausses.
- `option=noanalyzeprint` : Le texte automatique d'analyse de réponses n'est pas affiché. C'est donc à l'auteur de l'exercice de fournir un feedback convenable. Cela est utile dans certains types comme le type `mark` où le texte automatique est souvent inadéquat. Le score subsiste ainsi que l'indication sur la justesse de la réponse.
- A la suite d'une instruction `condition`, `option=hide` : La condition sera utilisée pour l'analyse de la réponse mais ne sera pas affichée.

D'autres options sont utilisables dans des types particuliers de réponses. Pour des précisions, revenir à la documentation du type de réponse.

Nom	Option
case	noreaccent
checkbox	color=" " eqweight nolegend shuffle show sort split
chemdraw	help image=" " match=" "

chessgame	color=" " noanswer
chset	norepeat
click	eqweight multiple nolegend shuffle sort split
clickfill	align=" " noorder transparent
clock	button=" " clocktype=" " init=" "
complex	absolute comma j precision
compose	linkword=" "
coord	feedback=" "
correspond	split
crossword	allhelp color=" " tooltip
dragfill	align=" " noorder transparent
draw	color=" " eqweight split
equation	eqsign=yes
flashcard	color=" " eqweight nolegend shuffle show sort split
function	integer
javacurve	precision=" "
jmolclick	eqweight split
jsxgraph	precision=" " color1=" " color2=" "
jsxgraphcurve	color1=" " color2=" " precision=" "
keyboard	keyboard=" "
mark	color=" " eqweight nolegend shuffle show sort split
matrix	split_coeff split_column split_row
menu	eqweight multiple nolegend shuffle sort split
multipleclick	color=" " eqweight nolegend shuffle show sort split
nocase	noreaccent
numeric	absolute comma nolegend
numexp	noreduction
puzzle	text
radio	eqweight multiple nolegend shuffle sort split
set	distinct_inputs repeat

Les tests de conditions

Dans certaines circonstances, il est souhaitable que les réponses libres soient testées par des conditions spéciales (questions avec plusieurs bonnes réponses, exercices demandant des exemples, réponses déterminées par des inégalités, etc). Ceci peut être réalisé de la façon suivante.

Si la bonne réponse à une question libre est une variable non définie auparavant, la réponse à cette question sera stockée dans cette variable. Cette variable peut ensuite être utilisée pour implémenter des conditions de test. Le score de l'exercice est alors calculé selon le nombre de conditions de test qui sont satisfaites, au lieu du nombre de réponses qui coïncident avec les bonnes réponses.

Jusqu'à 10 conditions simultanées peuvent être définies dans un exercice, sous l'implémentation actuelle. Seules les réponses numériques peuvent être ainsi testées. La syntaxe est (le dernier champ est optionnel)

```
\condition{texte d'affichage}{conditions de test}{weight=...}
```

Voici quelques exemples de conditions de test (`\rep1` et `\rep2` sont des variables non définies qui apparaissent dans des définitions de réponses libres, comme décrit ci-dessus.) Liste complète des relations

Condition	signification
`\rep1>=0 and \rep1<=1`	Bon si `rep1` est dans l'intervalle [0,1].
`\rep1=1 or \rep1=3 or \rep1=5`	Bon si `rep1` est 1, 3 ou 5.
`(\rep1)*(\rep2)=6 and (\rep1)+(\rep2)=5`	Bon si `rep1` et `rep2` sont les deux racines du polynôme x^2-5x+6.
`sin(\rep1)=0`	Bon si `rep1` est un multiple de π.

Voir aide sur le format oef pour la syntaxe de définition des réponses.

Commentaires ciblés selon les réponses

Vous pouvez définir des conditions (sur les réponses données par les utilisateurs) sous lesquelles des commentaires peuvent être imprimés dans la page de résultat d'un exercice. Une utilisation commune de cette facilité sera de donner des avertissements quand une erreur typique est détectée dans la réponse.

Un nombre arbitraire de commentaires ciblés peuvent être ajoutés, sur tout type de réponses (libre ou choix multiple). La syntaxe est comme suit.

```
\feedback{CONDITION}{MESSAGE}
```

Une telle ligne peut être placée dans le champ de paramètres (sous le mode guidé) ou n'importe où dans la source (sous le mode brut).

Quand `CONDITION` est remplie, `MESSAGE` sera montré à l'utilisateur dans la page de résultat. Les valeurs des réponses de l'utilisateur peuvent être utilisées dans les deux champs (CONDITION et MESSAGE) via les variables `\reply1`, `\reply2`, ... (pour réponses libres), `\choice1`, `\choice2`,... (pour choix multiples). L'ordre dans ces variables est le même que l'ordre sous lequel elles apparaissent dans la source (ou dans le formulaire en mode pas à pas).

`MESSAGE` est un texte libre, acceptant les tags html et les symboles mathématiques.

On pourra aussi s'aider de la variable `\sc_reply1`, `\sc_reply2`, ... dont la valeur vaut 1 si la réponse est bonne, 0 si la réponse est fausse et 0.5 si elle est bonne à précision près. Cette variable ne tient pas compte des conditions supplémentaires éventuellement imposées et est vide dans le cas où la réponse est uniquement analysée à l'aide de conditions.

Exemples de conditions Liste complète

condition	quand elle sera remplie
`\reply1=5`	la première réponse libre est exactement égale à 5
`\reply1>\reply2+2`	la première réponse libre est > la seconde plus 2
`\choice1=Oui and \choice2=Non`	premier choix multiple est Oui, et le second est Non (**attention** la comparaison dans les choix multiples est sensible aux minuscules-majuscules !)
`\reply1<5 or \reply1>8`	première réponse est hors de l'intervalle [5,8].
`3 isitemof \reply1`	la première réponse, du type `set` ou `vector`, contient une composante "3".

Bogue connu : Vous devez éviter le mélange de `\reply` et `\choice` dans une même condition. Sinon le résultat sera imprévisible.

Comment écrire les mathématiques dans votre exercice.

Vous pouvez inclure des symboles et formules mathématiques dans l'énoncé (quand il est sous format html), indication et solution.

Si vous voulez simplement insérer quelques symboles mathématiques ou caractères grecs, vous avez juste besoin de taper un `\` suivi du nom du symbole ou caractère grec. Par exemple, `\pi` vous donne π, `\le` donne $\leq$, `\pm` donne $\pm$, `\rightarrow` donne $\rightarrow$, etc. Les noms des symboles suivent la convention standard de TeX. En voici une liste (remplacez `$m_` par `\` dans les noms).

Plus généralement, pour insérer une formule mathematique entière, vous pouvez taper la formule de façon habituelle comme si vous l'entrez dans un logiciel mathématique ou une page d'outil de calcul sous WIMS. Pour que la formule soit formattée et montrée joliement, vous n'avez qu'à l'enfermer dans une paire de parenthèses précédées par le caractère \. Par exemple, `\(x^3-3x+cos(2pi*x)^5)` vous donne $x^3-3x+\cos(2\pi x)^5$, ou `\(sqrt(x^2+y^2))` vous donne $\sqrt{x^2+y^2}$. Vous pouvez taper `\(integrate(x^2+1)dx)` pour $\int (x^2+1)\mathrm{dx}$, ou `\(integrate(exp(x^2+1),x=1..infinity))` pour $\int_1^\infty \exp(x^2+1)\mathrm{dx}$. Vous pouvez aussi écrire la somme $\sum_{n=1}^\infty \frac{1}{n^2}$ via `\(sum(1/n^2,n=1..infinity)`, ou le produit $\prod_{n=1}^\infty \frac{n}{n+1}$ via `\(product(n/(n+1),n=1..infinity)`.

D'autres aides sur la manière d'écrire des expressions mathématiques sont dans cette page.

Pour montrer une matrice $\begin{pmatrix}1&2&3\\4&5&6\\7&8&9\end{pmatrix}$, vous devez taper `\([1,2,3;4,5,6;7,8,9])`. Des matrices dans les matrices sont acceptées (et seront montrées correctement).

Pour les experts de TeX ou LaTeX, notez que quand le logiciel voit une formule enfermée dans une paire de parenthèses précédées par `\`, qui contient des commandes spécifiques à TeX ou LaTeX, il va l'interpréter en tant que source TeX, et donc va l'envoyer directement à TeX pour formatage. Ceci vous permet d'écrire des formules très sophistiquées si vous savez comment les écrire en TeX. Dans le cas où vous voulez forcer une formule à être une image produite par TeX ou LaTeX, écrivez dans les parenthèses `\displaystyle` ou une paire d'accolades vide `{}`. Par exemple, vous obtiendrez a via `\(\displaystyle a )` ou `\({} a )` et a via `\( a )`.

La façon la plus complète d'utiliser TeX dans votre exercice peut être de choisir le format TeX (pour l'énoncé seulement). Mais vous n'avez alors le droit à aucune commande html.

Dessins dynamiques

Il est possible d'inclure des dessins 2D dans vos documents. Ces dessins sont dynamiques, c'est-à-dire vous pouvez utiliser des variables dans vos commandes de dessin, et le résultat variera avec les valeurs des variables. Il est même possible de produire des animations.

Voir aussi canvasdraw pour des possibilités d'interaction plus grandes.

Syntaxe:

```
\draw{xsize,ysize}{commande...}
```

où `xsize, ysize` sont respectivement la largeur et la hauteur du dessin (en nombre de pixels), et `commande...` est le contenu des commandes. Cela est du texte composé de plusieurs lignes, chaque ligne étant une commande.

Exemple.

Chaque commande prend une ligne.

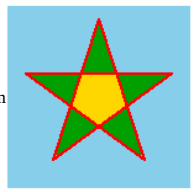
Dans la table suivante `[color]` peut être un nom de couleur ou 3 entiers 0 entre 255, séparés par des virgules, correspondant aux valeurs de rouge, vert, bleu.

Commandes	
Nom - Syntaxe	
Synonymes	**Signification**
`affine a,b,c,d,tx,ty`	
	Applique la transformation affine (x ;y) -> [a,b ;c,d](x ;y)+(tx ;ty) aux objets définis ultérieurement.
`animate fra,del,rep`	
	Cette commande est seulement valable pour les exercices OEF et les documents. Elle doit apparaître sur la première ligne de commande. Il s'agit d'une animation de l'image de fra plans, d'intervalle de del secondes, rep fois; si rep=0, répétition en boucle infinie.
`animstep n`	
	Set up an integer which can be called in any evaluation. Used for animation. Direct use of this command must be avoided under WIMS.
`arc x,y,w,h,a1,a2,[color]`	
	Arc de l'ellipse de largeur w et de hauteur h centrée en (x,y) (coordonnées mathématiques) entre l'angle a1 et l'angle a2 en degrés.
`arrow x1,y1,x2,y2,l,[color]`	
	Flèche allant du point (x1; y1) vers le point (x2; y2) et dont la tête est de longueur l pixels.
`arrow2 x1,y1,x2,y2,l,[color]`	
	Flèche entre les points (x1; y1) et (x2; y2) ayant deux têtes de longueur l pixels.
`circle x,y,d,[color]`	
	Cercle de centre (x; y) et de diamètre d pixels.
`comment`	
	Ligne de commentaire.
`copy x,y,x1,y1,x2,y2,[filename]`	
`insert`	Insère le rectangle de diagonale (x1; y1) et (x2; y2) (dans le repère en pixels) du fichier filename au point (x; y) : l'extrémité en haut à gauche de l'image est au point (x; y). Si x1 = y1 = x2 = y2 = -1, tout le fichier est copié. [filename] est l'adresse du fichier à partir du répertoire wims/public_html/gifs ou du répertoire indiqué dans common_images pour les modules OEF.
`copyresized x1,y1,x2,y2,dx1,dy1,dx2,dy2,[filename]`	
	Insère le rectangle de diagonale (x1; y1) et (x2; y2) du fichier filename dans le rectangle de diagonale (dx1; dy1) et (dx2; dy2) (remise à la taille réalisée). si x1 = y1 = x2 = y2 = -1, tout le fichier filename est copié
`crosshair x1,y1,[color]`	
	dessine une croix au point (x1,y1)
`crosshairs [color], x1,y1,x2,y2,...`	
	dessine des croix aux points de coordonnées (x1,y1), (x2,y2), ...
`crosshairsize w`	
	Règle la taille des croix à w (en pixels).
`darrow x1,y1,x2,y2,l,[color]`	
`dasharrow dashedarrow`	Flèche en pointillés allant du point (x1; y1) vers le point (x2; y2) dont la tête est de longueur l pixels.
`darrow2 x1,y1,x2,y2,l,[color]`	
`dasharrow2 dashedarrow2`	Flèche en pointillés entre les points (x1; y1) et (x2; y2) et à deux têtes de longueur l pixels.
`dhline x,y,[color]`	
`dashedhorizontalline dashhorizontalline hdline horizontaldashedline`	Droite horizontale en pointillés passant par le point (x; y).
`diamondfill x,y,nx,ny,[color]`	
`diafill`	Remplit la région contenant le point (x; y) avec des lignes de couleur color (quadrillage oblique). (nx; ny) est la distance verticale et horizontale (en pixels) entre deux lignes.
`dline x1,y1,x2,y2,[color]`	
`dashedline dashline`	Segment en pointillés entre les points de coordonnées (x1; y1) et (x2; y2).
`dlines [color],x1,y1,x2,y2,x3,y3...`	
`dashedlines dashlines`	Ligne polygonale en pointillés joignant les points (x1; y1), (x2; y2), (x3; y3) ...
`dotfill x,y,nx,ny,[color]`	
`pointfill diskfill`	Remplit la région contenant le point (x; y) avec des gros points de couleur color. (nx; ny) est la distance verticale et horizontale entre deux points.
`dvline x,y,[color]`	
`dashedverticaline dashverticalline vdline verticaldashedline`	Droite verticale en pointillés passant par le point (x; y).
`ellipse x,y,w,h,[color]`	
	Ellipse de largeur w et de hauteur h centrée en (x,y).
`fcircle x,y,d,[color]`	

ball disk filledcircle	Disque de centre (x; y) et de diamètre d pixels.
fellipse x,y,w,h,[color]	
filledellipse	Ellipse de largeur w et de hauteur h centrée en (x,y) et remplie avec la couleur color.
fill x,y,[color]	
flood floodfill	Colorie la région contenant le point (x; y) avec la couleur color
filltoborder x,y,[color1],[color2]	
	Colorie avec la couleur color2 la région contenant (x; y) et délimitée par la couleur color1.
fpoly [color],x1,y1,x2,y2,x3,y3...	
filledpoly filledpolygon fpolygon	Polygone de sommets (x1; y1), (x2; y2), (x3; y3) ... et rempli avec la couleur color
frect x1,y1,x2,y2,[color]	
filledrect fillecrectangle frectangle	Rectangle de diagonale (x1; y1) et (x2; y2) et rempli avec la couleur color.
fsquare x,y,s,[color]	
filledsquare	Carré de coin supérieur gauche (x; y) et de côté de longueur s, rempli avec la couleur color.
ftriangle x1,y1,x2,y2,x3,y3,[color]	
filledtriangle	Triangle de sommets (x1; y1), (x2; y2), (x3; y3) et rempli avec la couleur color.
gridfill x,y,nx,ny,[color]	
	Remplit la région contenant le point (x; y) avec des lignes de couleur color (quadrillage droit) . (nx; ny) est la distance verticale et horizontale entre deux lignes.
hatchfill x,y,nx,ny,[color]	
	Remplit la région contenant le point (x; y) avec des lignes (simples) de couleur color. (nx; ny) est la distance verticale et horizontale entre deux lignes.
hline x,y,[color]	
horizontalline	Droite horizontale passant par le point (x; y).
interlace	
	Set interlaced image
killaffine	
	Réinitialise la transformation affine à l'identité.
killbrush	
	Turns off brush selection for line drawing.
killlinear	
killrotation,killrotate	Réinitialise la transformation linéaire à l'identité.
killtile	
	Désactive la sélection de pavage pour le remplissage.
killtranslation	
killtranslate	Réinitialise la translation au vecteur nul.
lattice x0,y0,x1,y1,x2,y2,n1,n2,[color]	
	Réseau de n1xn2 points partant de (x0,y0), avec n1 lignes dans la direction de (x1,y1) et n2 colonnes dans la direction de(x2,y2).
levelcurve [color],[expression],l1,l2,...	
	Dessine des courbes de niveau de la surface décrite par une expression de niveaux l1, l2,...
levelstep n	
	Règle le nombre d'étapes en pixels utilisé pour le dessin des courbes de niveaux. Entre 1 and 16, defaut : 4.
segment x1,y1,x2,y2,[color]	
seg line (deprecated)	Segment entre les points de coordonnées (x1; y1) et (x2; y2).
linear a,b,c,d	
	Applique la transformation linéaire (x;y) -> [a,b;c,d](x;y) aux objets définis ultérieurement..
polyline [color],x1,y1,x2,y2,x3,y3...	
lines (deprecated)	Ligne polygonale joignant les points (x1; y1), (x2; y2), (x3; y3) ...
linewidth w	
	Epaisseur des traits à w pixels.
multicopy n1,n2,...,nk, [filename]	
	Copie l'image [filename] dans le parallélogramme donné par la commande setparallelogram et applique à l'image les transformations n1, ..., nk (au maximum 19). Si n1 ... ne sont pas donnés, toutes les transformations définies précédemment sont appliquées. Attention, setparallelogram et au moins un setvector ou setmatrix ou settransform doivent d'abord avoir été définis.
new x,y	
	Fait une nouvelle image de taille x,y (en pixels).

output [filename]	
	Sauve l'image dans le fichier [filename].
parallel x1,y1,x2,y2,xv,yv,n,[color]	
	n segments parallèles partant du segment d'extrémités (x1; y1) et (x2; y2) avec le déplacement de vecteur (xv; yv).
pixels [color],x1,y1,x2,y2,...	
	Points de diamètre 1 aux coordonnées (x1; y1), (x2; y2), ...
plot [color],[formula]	
curve	Courbe représentative de la fonction formula.
plotjump j	
	Saut de la courbe tracée si deux points consécutifs ont une distance de plus de j pixels. Utile afin d'éviter de dessiner des fonctions discontinues comme des fonctions continues. Valeur par défaut : 200.
plotstep n	
plotsteps tstep tsteps	Nombre de points calculés dans le tracé de courbes. Valeur par défaut : 100.
point x,y,[color]	
	Point de coordonnées (x; y) et de diamètre l'épaisseur de trait.
points [color],x1,y1,x2,y2,...	
	Points de coordonnées (x1; y1), (x2; y2), ... et de diamètre l'épaisseur de trait.
polygon [color],x1,y1,x2,y2,x3,y3...	
poly	Polygone de sommets (x1; y1), (x2; y2), (x3; y3)...
range x1,x2,y1,y2	
	Détermine les coordonnées des bords de l'image.
rays [color],x0,y0,x1,y1,x2,y2...	
	Segments joignant (x0; y0) et (x1; y1), (x0; y0) et (x2; y2), ...
rect x1,y1,x2,y2,[color]	
rectangle	Rectangle de diagonale (x1; y1) et (x2; y2).
resetmatrix n	
	Réinitialise la n-ième transformation linéaire à l'identité.
resetparallelogram	
	Réinitialise setparallelogram
resetvector n	
	Réinitialise la n-ième translation à l'identité.
resettransform n	
	Réinitialise la n-ième transformation linéaire ou affine ou la n-ième translation aux valeurs initiales (identité ou vector nul).
rotation d	
rotate	Rotation des objets définis ultérieurement de d degrés dans le sens inverse des aiguilles d'une montre, de centre (0;0)
setbrush [filename]	
	Utilise l'image [filename] comme "brush" pour tous les dessins de ligne.
setmatrix n,a,b,c,d	
	Définit la transformation linéaire pour les objets de multicopy (x;y) -> [a,b;c,d](x;y).
setparallelogram xs,ys,xu,yu,xv,yv	
	Prépare l'endroit où l'image sera copiée par multicopy (coordonnées mathématiques) : xs,ys coordonnées mathématiques de l'origine, xu,yu coordonnées mathématique de la "ligne horizontale de l'image" à copier, xv,yv coordonnées mathématique de la "ligne verticale de l'image" à copier.
setpixel x,y,[color]	
	Point de coordonnées (x; y) et de diamètre 1.
setstyle [color1],[color2],...	
	Définit la couleur des lignes the line style comme color1,color2,...
settile [filename]	
	Utilise l'image [filename] comme modèle pour toutes les commandes de remplissages.
settransform n,a,b,c,d,tx,ty	
	Définit la n-ième transformation linéaire pour les objets de multicopy (x;y) -> [a,b;c,d](x;y) + (tx;ty).
setvector n,tx,ty	
	Définit la n-ième translation transformation linéaire pour les objets de multicopy: (x;y) -> (tx,ty).
size x,y	
	Set the image size to x pixels horizontally and y pixels vertically.
square x,y,s,[color]	

	Carré de coin supérieur gauche (x; y) et de côté s (en pixels).
`text [color],x,y,[font],[string]`	
`print string write`	Ecrit string au point de coordonnées (x; y) avec la police font=small,medium,large ou giant.
`textup [color],x,y,[font],[string]`	
`stringup writeup`	Ecrit string de bas en haut au point de coordonnées (x; y) avec la police font=small,medium,large ou giant.
`trange t1,t2`	
`ranget`	Intervalle du paramètre pour le tracé des courbes paramétriques (par défaut 0 et 1).
`translation tx,ty`	
`translate`	Applique la translation (x;y) -> (x;y)+(tx;ty) aux objets définis ultérieurement.
`transparent [color]`	
	Définit la couleur color comme transparente.
`triangle x1,y1,x2,y2,x3,y3,[color]`	
	Triangle de sommet (x1; y1), (x2; y2), (x3; y3).
`vimg n`	
	Active (1) ou désactive (0) la sortie en graphique vectorielle (défaut 0)
`vimgfile [filename]`	
	Sortie en graphique vectorielle (pour l'instant seulement DXF) dans le fichier [filename].
`vline x,y,[color]`	
	Droite verticale passant par le point (x; y).
`xrange x1,x2`	
`rangex`	Détermine les coordonnées horizontales mathématiques des bords de l'image.
`yrange y1,y2`	
`rangey`	Détermine les coordonnées verticales mathématiques des bords de l'image.

Exemple du dessin dynamique Le dessin  est créé par la commande suivante.

```
\draw{150,150}{xrange -1.2,1.2
yrange -1.2,1.2
fill 0,0,skyblue
linewidth 2
polygon red,0,1,sin(4*pi/5),cos(4*pi/5),sin(8*pi/5),cos(8*pi/5),sin(2*pi/5),cos(2*pi/5),sin(6*pi/5),cos
fill 0,0,gold
fill 0,0.5,green
fill 0.5*sin(2*pi/5),0.5*cos(2 *pi/5),green
fill 0.5*sin(4*pi/5),0.5*cos(4*pi/5),green
fill 0.5*sin(6*pi/5),0.5*cos(6*pi/5),green
fill 0.5*sin(8*pi/5),0.5*cos(8*pi/5),green
}
```

syntaxe des commandes

Possibilités avancées de OEF

Cacher le nom d'images

Si l'exercice demande de reconnaître une image prise au hasard, le nom de l'image peut donner une indication à l'exercice. Pour cacher ce nom d'image, vous pouvez écrire

```
\img{\imagedir/picture.jpg alt="picture"}
```

Ainsi dans le source de la page html, l'étudiant pourra voir

```
<img src="a_random_unrelated_file_name" alt="picture" />
```

Le vrai nom de l'image est ainsi caché.

Remarque. N'utilisez pas directement la commande WIMS *rename* pour traiter le nom de fichier, sinon l'exercice ne fonctionnera pas correctement dans l'environnement sauvegardé. D'autre part, le nom sera changé uniquement dans le cas où l'exercice est sauvé dans un module et non dans une classe.

Exercices à plusieurs étapes

Des exercices à plusieurs étapes peuvent être conçus grâce à la commande `\steps` . Par exemple, si vous définissez (parmi d'autres paramètres)

```
\steps{choice 1, reply 1
choice 2, reply 2, reply 3
choice 3
}
```

l'exercice sera présenté en trois étapes, la première proposant un choix multiple (choice 1) et une réponse de type `reply` (reply 1), la seconde proposant un choix multiple et deux réponses de type `reply` , etc.

L'argument de `\steps` accepte des paramètres définis auparavant aussi bien que des définitions conditionnelles de la même manière que pour la définition de paramètres. Cela permet de présenter des champs de réponses sélectivement selon des variations aléatoires de paramètres, même dans le cas où il n'y a qu'une seule étape.

Il faut noter que si une erreur est faite à une étape, les étapes suivantes ne sont pas proposées et sont donc considérées comme fausses.

Le numéro de l'étape en cours est accessible dans le champ `statement` dans la variable `\step` qui prend donc successivement les valeurs 1,2,... à mesure que l'utilisateur avances dans ses réponses.

Pour faire des exercices dynamiques à plusieurs étapes (étapes variant selon les réponses de l'étudiant), vous pouvez utiliser la commande `\nextstep{...}` . L'argument de `\nextstep` est semblable à celui de `\steps` mais seule la première ligne est effective. Elle n'est utile que si cet argument est une variable qui change après avec les définitions de paramètres après soumission. L'exercice s'arrête quand le contenu de `\nextstep` est vide.

Dans tous les cas, une seule déclaration parmi `\steps` ou `\nextstep` doit apparaître dans un source OEF.

Branchements conditionnels et boucles

Commandes disponibles:

```
\if{condition}{contenu conditionnel}
\if{condition}{contenu_conditionnel}{autre_contenu_conditionnel}
\ifval{condition}{contenu conditionnel}
\ifval{condition}{contenu_conditionnel}{autre_contenu_conditionnel}
\for{var=n1 to n2}{contenu_de_la_boucle}
\while{condition}{contenu_de_la_boucle} (uniquement dans l'environnement principal)
```

Les commandes de branchements conditionnels et de boucles peuvent être utilisées dans l'environnement principal ou (sauf pour `while`) à l'intérieur de l'énoncé (`statement`), d'une aide (`hint`), d'une solution (`solution`), d'un feedback (`feedback`). Lorsque ces commandes sont dans l'environnement principal, elles n'affectent que la définition de paramètres.

Méthodes spéciales

La commande `\special{...}` dans le corps d'un exercice (`statement`) permet de faire les méthodes suivantes :

La méthode spéciale `imagefill`

Syntaxe générale : `\special{imagefill` *paramètres*`}`
Met les champs des types de réponses `dragfill` ou `clickfill` dans une grande image. Doit être utilisé dans le `statement` d'un exercice OEF. Exemple

```
\special{imagefill \imagedir/myphoto.jpg,450x350,40x40
reply 1,120x250
reply 3,300x50
reply 4,10x15
}
```

Dans cet exemple, on affiche une grande image (**`\imagedir/myphoto.jpg`**) dans l'énoncé de type 450x350, avec 3 champs de réponse de type drag-and-drop de type 40x40. Les trois champs sont respectivement les réponses 1, 3 et 4 (qui doivent être de type `clickfill` ou `dragfill`), aux positions respectives 120x250, 300x50, 10x15 dans la grande image.

La grande image sera redimensionnée à la taille spécifiée.

Il est possible d'avoir des réponses multiples. Pour cela, ajouter un troisième paramètre entier à la taille : `reply1,120x250x4` montrera un champ de longueur 4 (x 120).

Avec l'option `transparent` dans le type de réponse, le champ à remplir est transparent et ne cache pas l'image.

La méthode spéciale `codeinput`

Syntaxe générale : `\special{codeinput` *paramètres*`}`
Met les champs des types de réponses de type texte dans un texte en gardant sa présentation (espaces, retours à la ligne, ...). Cela est utile si l'on désire faire remplir des trous dans un code de programmation. Doit être utilisé dans le `statement` d'un exercice OEF.

La première ligne est formée du texte du code mis entre crochets, suivi de la taille générique des champs de réponse (obligatoire). Les lignes suivantes sont formées des réponses insérées suivies si nécessaires de la taille de cette réponse si elle est différente de la taille générique. Les endroits d'insertion de la réponse sont marqués par le mot reply n .

Exemple

```
\statement{
\special{codeinput [texte], 5
reply1,2x8
reply3
reply4,1x10
}
```

Dans l'exemple ci-dessus, on affiche le code dans l'énoncé avec 3 champs de réponse. Les trois champs sont respectivement les réponses 1, 3 et 4 (qui peuvent être toute réponse faisant intervenir un champ de texte comme `numeric` , `case formal` , etc.)

```
\text{code=for i = 1 to reply1
  do
    reply2
  od
}
\statement{
 \special{codeinput [\code], 5
reply1
reply2,30
 }
}
\answer{}{4}{type=numeric}
\answer{}{tourner d'un quart de tour}{type=nocase}
```

Option: Il est possible de rajouter un mot d'option, pour l'instant uniquement `div`. Dans ce cas, le texte ne sera pas affiché dans un style `pre`.

Attention : il peut y avoir un bogue pour l'instant dans le cas où un autre champ de réponses se trouve sur la page (numérotation dans l'analyse) que l'on espère corriger.

La méthode spéciale `imageinput`

Syntaxe générale : `\special{imageinput` *`paramètres`*`}`
Met des champs de types de réponses texte sur une image. Doit être utilisé dans le `statement` d'un exercice OEF.

La première ligne est formée de l'adresse de l'image, suivie de la taille générique des champs de réponse (obligatoire). Les lignes suivantes sont de la forme

```
reply n, p_X x p_Y,size, css
```

où `p_X` et `p_Y` donnent la position en pixels du coin en haut à gauche du champ correspondant, `size` est la taille du champ (facultatif), `defaut` est un texte s'affichant dans le champ correspondant (facultatif), `css` un code css (facultatif).
Exemple

```
\statement{
 \special{imageinput image_url, 200x200,5
 reply 1,100x100
 reply 2,150x150
 }
}
\answer{}{x^2}{type=formal}
\answer{}{2}{type=numeric}
```

```
\text{D=draw(200,200
xrange -5,5
yrange -5,5
hline 0,0,black
vline 0,0,black)}
\statement{
\special{imageinput \D, 200x200,3
reply 1,180x80
reply 2,102x0}
}
\answer{}{x}{type=case}
\answer{}{y}{type=case}
```

Attention : il peut y avoir un bogue pour l'instant dans le cas où un autre champ de réponses se trouve sur la page (numérotation dans l'analyse) que l'on espère corriger.

La méthode spéciale `mathmlinput`

Syntaxe générale : `\special{mathmlinput` *`paramètres`*`}`
Met des champs de types de réponses texte dans une formule mathématique. Doit être utilisé dans le `statement` d'un exercice OEF.

La première ligne est formée de la formule mathml mise entre crochets, suivie de la taille générique des champs de réponse (obligatoire). Les endroits d'insertion de la réponse sont marqués dans la formule mathématique par le mot reply `n`. Les lignes suivantes sont de la forme

```
reply n, size, css
```

où `size` est la taille de la réponse (facultatif), `css` un code css (facultatif). Exemple

```
\statement{
 \special{mathmlinput [\frac{reply1}{reply2}],5,noanswer
 reply 1
 reply 2
 }
}
\answer{}{x^2}{type=formal}
\answer{}{2}{type=numeric}
```

Dans cet exemple, on affiche une fraction dans l'énoncé avec 2 champs de réponse, l'un pour le numérateur, l'autre pour le dénominateur. Les deux champs sont respectivement les réponses 1, 2 (qui peuvent être toute réponse faisant intervenir un champ de texte comme `numeric`, `case formal`, etc.) La taille des champs est de 5 caractères

```
\statement{
\special{mathmlinput [\frac{reply1}{reply2}],4
reply 1,10,color:blue;font-size:20px
reply 2}
}
\answer{}{x^2}{type=formal}
\answer{}{2}{type=numeric}
```

Dans cet exemple, la taille du premier champ est de 10 caractères, ce qui est répondu apparait en bleu et de taille 20 pixels.

Lorsqu'une des réponses est fausse, la réponse bien formatée est affichée à côté de la formule dans le texte. Si on ne désire pas la voir apparaitre, rajouter l'item `noanswer` sur la première ligne, après la taille générique.

```
\statement{
\special{mathmlinput [\frac{reply1}{reply2}],4,noanswer
reply 1,10,color:blue;font-size:20px
reply 2}
}
\answer{}{x^2}{type=formal}
\answer{}{2}{type=numeric}
```

Attention : il peut y avoir un bogue pour l'instant dans le cas où un autre champ de réponses se trouve sur la page (numérotation dans l'analyse) que l'on espère corriger.

La méthode spéciale `expandlines`

Syntaxe générale : `\special{expandlines` *`paramètres`*`}`

Ecrit le paramètre dans un style `pre` sans évaluation.

Exemple

```
\text{texte=
for a in \B
   a = a + 1
endfor
}
```

La méthode spéciale `help`

Syntaxe générale : `\special{help` *`paramètres`*`}`

La méthode spéciale `help` accepte deux paramètres. Le premier est un identificateur qui sera dans la variable `\help_subject` à l'intérieur de la commande `\help` et le second est le texte du lien. La classe css du lien est `oef_specialhelp` et peut être configurée. Si `\special{help}` est mis dans l'énoncé ou le feedback d'un exercice, l'aide sera de type popup, contrairement à l'aide usuelle. Tout le contenu de l'aide doit être mis dans le source de l'exercice oef. Aucune facilité n'est prévue pour l'aide au niveau du module, car le principe est que l'indépendance des fichiers oef n e doit pas être détruit par les aides. Cependant, si vous avez plusieurs exercices partageant les mêmes textes d'aide vous pouvez utiliser cpp.

Exemple

```
 \title{Deviner}
 \statement{ Deviner un mot :
   \special{help test1,Premier indice}
 }
 \answer{}{Singe}
 \help{
  \if{\help_subject issametext }{
     Cliquer sur le bouton à l'intérieur de l'énoncé pour voir le premier indice.
   }
   \if{\help_subject issametext test1}{
     L'indice est S.
   }
 }
```

Voici un exemple d'aides imbriquées.

```
\title{Deviner}
\statement{ Deviner un mot :
  \special{help test1,Premier indice}
}
\answer{}{Singe}
\help{
  Voici l'aide: help subject=\help_subject
  \if{\help_subject issametext}{
    \special{help test0,ici}
  }
  \if{\help_subject issametext test1}{
    La première lettre est <span class="wims_emph">S</span>. Vous pouvez trouver un nouvel indice
    \special{help test2,ici}
  }
 \if{\help_subject issametext test2}{Voici le second indice.
    La deuxième lettre est <span class="wims_emph">i</span>
  }
 \if{\help_subject issametext test0}{Voici
   \special{help test1,l'aide 1}, \special{help test2,l'aide 2}
  }
}
```

La méthode spéciale `rename`

Syntaxe générale : `\special{rename paramètres}`
Renomme un fichier (image essentiellement).

La méthode spéciale `tooltip`

Syntaxe générale : `\special{tooltip paramètres}`
crée une aide popup (tooltip) : le premier paramètre est le texte du lien, le troisième paramètre est le texte apparaissant dans l'aide popup. Le second paramètre est l'option entre crochets : entre les crochets, vous pouvez mettre l'option comme dans la documentation de wz_tooltip.js. Par défaut : `[FONTSIZE, '12pt', ABOVE, 'true']` Si vous voulez ne charger qu'une seule fois le javascript (une fois est suffisant pour une page html), ne mettez aucun paramètre. Si vous ne voulez pas charger le javascript car il a été déjà chargé dans la page html, ajoutez `nojs` au second paramètre. La classe css du lien est `span.tooltip`. Exemple

```
\special{tooltip passer la souris ici, ,le mot a 5 lettres}
\special{tooltip passer la souris ici,[DURATION, 4000, FONTSIZE, '18pt'], le mot a 5 lettres}
\special{tooltip passer la souris ici,[TITLE, 'Some Title', PADDING, 9],un texte}
\special{tooltip }
\special{tooltip passer la souris ici,nojs [TITLE, 'Some Title', PADDING, 9],un texte}
```

Les options possibles sont (from www.walterzorn.com)

Command	Description
ABOVE	Positions the tooltip above the mousepointer. Value: true or false. Combine with OFFSETY to adjust the vertical distance from the mousepointer, or with CENTERMOUSE to center the tooltip horizontally above the mousepointer.

BGCOLOR	Background color of the tooltip. Value: HTML color, in single quotes, e.g. '#D3E3F6' or 'DarkCyan', or empty string '' for no background.
	Example:
	`onmouseover="Tip('Some text', BGCOLOR, '#D3E3F6')"`
	or
	`onmouseover="Tip('Some text', BGCOLOR, '')"`

BGIMG	Background image. Value: Path to image, in single quotes. Example: `onmouseover="Tip('Some text', BGIMG, '../images/tipbackground.gif')"`
BORDERCOLOR	Border color. Value: HTML color, in single quotes, e.g. '#dd00aa'.
BORDERSTYLE	Border style. Value: CSS border style, in single quotes. Recommend are 'solid' (default), 'dashed' or 'dotted', others may not work in all browsers.
BORDERWIDTH	Width of tooltip border. Value: Integer ≥ 0. Default is 1. Use 0 for no border. Example: `onmouseover="Tip('Some text', BORDERWIDTH, 2)"`
CENTERMOUSE	Centers the tooltip horizontally beneath (or above) the mousepointer. Value: true or false. Consider that the tooltip is offset from the center by the value globally set in wz_tooltip.js (`config. OffsetX`), or as specified by the OFFSETX command. Example: `onmouseover="Tip('Some text', CENTERMOUSE, true, OFFSETX, 0)"`
CLICKCLOSE	Closes the tooltip once the user clicks somewhere inside the tooltip or into the document. Value: true, false.
CLOSEBTN	Displays a closebutton in the titlebar. Value: true, false.
CLOSEBTNCOLORS	Colors used for the closebutton. Value must be a comma-separated array of 4 color values. Each color in single quotes. The entire array must be enclosed with a pair of square brackets, see example below, since it's actually a single parameter. The 4 colors have the following meanings: 1. Background color 2. Text color 3. Highlighted background, while the button is being hovered 4. Hilighted text color, while the button is being hovered

For each of these colors, you can also specify an empty string `''`, in which case the title background, or title text color, respectively, is used for that button state.

Example:
`Tip('Text', CLOSEBTN, true, CLOSEBTNCOLORS, ['', '#66ff66', 'white', '#00cc00'], STICKY, true)`
In this example, the first color value (background color) is an empty string. Therefore the closebutton inherits the titlebar background.

CLOSEBTNTEXT — Text in the closebutton. Value must be enclosed with single quotes. Example:
`Tip('Tooltip text', CLOSEBTN, true, CLOSEBTNTEXT, 'Click Me', STICKY, true)`
Globally preset in wz_tooltip.js is ' X ' - the whitespace entities ' ' add some horizontal padding to the button.

COPYCONTENT — COPYCONTENT has only effect on tooltips created with `TagToTip()`, that is, if an HTML element is to be converted to a tooltip.
Value: true, false.
If true (this is the default behaviour preset in wz_tooltip.js), just a copy of the text (or inner HTML) of the HTML element is inserted into the tooltip. If false, the entire HTML element (its DOM node) by itself is temporarily converted to a tooltip, which may be useful in the following aspects:
1.) If the HTML element converted to a tooltip contains a form with inputs, their current user input will be maintained even while the tooltip is not displayed.
2.) The tooltip inherits the style properties of the HTML element.
Example how to convert an HTML element entirely to a tooltip, by applying COPYCONTENT with the value `false`:
`TagToTip('SomeID', COPYCONTENT, false, BORDERWIDTH, 0, PADDING, 0)`
Moreover, this example turns off the native tooltip border (BORDERWIDTH, 0), and sets the native tooltip padding to zero, so only the padding and border defined for the HTML element itself are displayed.

DELAY — Tooltip shows up after the specified timeout, in milliseconds. A behavior similar to OS based tooltips.
Value: Integer ≥ 0. Use 0 for no delay. In wz_tooltip.js preset and recommended is 400.
Example:
`onmouseover="Tip('Some text', DELAY, 1000)"`

DURATION — Time span, in milliseconds, until the tooltip will be hidden, even if the `STICKY` command has been applied, or even if the mouse is still on the HTML element that has triggered the tooltip.
Value: Integer ≥ 0. Use 0 for no limitation (this is the default)

FADEIN — Fade-in animation. The value (integer ≥ 0) specifies the duration of the animation, in milliseconds. 0 for no animation.

Not supported in Opera prior to v.9.0, old versions of Safari, some versions of Konqueror. These fall back to normal, non-animated behaviour.

FADEOUT — Fade-out animation. The value (integer ≥ 0) specifies the duration of the animation, in milliseconds. 0 for no animation.

Recommended: combine with FADEIN.

Not supported in Opera prior to v.9.0, old versions of Safari, some versions of Konqueror. These fall back to normal, non-animated behaviour.

FIX — Shows the tooltip at the fixed coordinates [x, y]. Value: Square-bracketed array of two integers. Example:
`onmouseover="Tip('Some text', FIX, [230, 874])"`
You can also call function(s) defined elsewhere that calculate the coordinates dynamically:
`onmouseover="Tip('Text', FIX, [CalcFixX(), CalcFixY()], BORDERWIDTH, 2)"`
or
`onmouseover="Tip('Text', FIX, CalcFixXY(), ABOVE, true)"`
In the latter example, the function CalcFixXY() must return an array containing the two numbers.

FOLLOWMOUSE — The tooltip follows the movements of the mouse. Value: true, false. Default: true.

When turning this off by applying the value false, the tooltip behaves like OS-based tooltips which don't follow the mouse.

FONTCOLOR — Font color. Value: HTML color, in single quotes, e.g. '#990099'

FONTFACE — Font face (font family).
Value: As you'd specify it in HTML or CSS, enclosed with single quotes, e.g. `Tip('Some text', FONTFACE, 'Arial, Helvetica, sans-serif', FONTSIZE, '12pt')`

FONTSIZE — Font size. Value: Size with unit, in single quotes. Unit ('px', 'pt', 'em' etc.) is mandatory.

FONTWEIGHT — Font weight. Value: 'normal' or 'bold', in single quotes.

LEFT — Tooltip positioned on the left side of the mousepointer. Value: true, false.
LEFT and ABOVE commands combined.
Example:
`onmouseover="Tip('Some text', LEFT, true, ABOVE, true)"`

OFFSETX — Horizontal offset from mouse-pointer. Value: Any integer. May also be negative.

OFFSETY — Vertical offset from the mouse-pointer. Value: Any integer. May also be negative.

OPACITY — Transparency of tooltip. Value: Integer between 0 (fully transparent) and 100 (opaque, no transparency).

Opacity is the opposite of transparency, i.e.
opacity = 100 - transparency (in percent).
Another example with image (taken on my 9000-km / 5500-miles recumbent bicycle trip Hamburg-Northcape-Munich), shadow via SHADOW command, content centered using TEXTALIGN, background image via BGIMG and animation via FADEIN and FADEOUT commands.

Not supported in Opera prior to v.9.0, old versions of Safari and some versions of Konqueror.

PADDING — Inner spacing of tooltip, between border and content. Value: Integer ≥ 0.

SHADOW	Tooltip drops a shadow. Value: true, false
SHADOWCOLOR	Shadow color. Value: HTML color, in single quotes. Example: `onmouseover="Tip('Some text', SHADOW, true, SHADOWCOLOR, '#dd99aa')"`
SHADOWWIDTH	Shadow width (offset). Value: Integer ≥ 0. Example: `onmouseover="Tip('Some text', SHADOW, true, SHADOWWIDTH, 7)"`
STICKY	The tooltip stays fixed at its initial position until another tooltip pops up. Value: true, false. With DURATION you can enforce the tooltip to disappear after a certain time span.
TEXTALIGN	Aligns the text in the body of the tooltip. Value: 'right', 'center', 'justify' or 'left'. Example: `onmouseover="Tip('Some text', TEXTALIGN, 'right')"`
TITLE	Display a titlebar. Value: Text to display, in single quotes. May even contain HTML, which gives you additional freedom in fashioning the titlebar.
TITLEALIGN	Aligns the text in the titlebar. Value: 'right', 'center', 'justify' or 'left'
TITLEBGCOLOR	Backgroundcolor of the titlebar. Value: HTML color, in single quotes. If it's an empty string `''`, the border color (see also BORDERCOLOR command) is used (this is the default).
TITLEFONTCOLOR	Color of title text. Value: HTML color, in single quotes. If it's an empty string `''`, the backgroundcolor of the tooltip body (see also BGCOLOR command) is used (this is the default).
TITLEFONTFACE	Font face for title text. Value: Like in HTML or CSS. Enclosed with single quotes. If the value is an empty string `''`, the tooltip body font, in boldified form, is used (this is the default). Example: `onmouseover="Tip('Some text', TITLE, 'Some Title', TITLEFONTFACE, 'Verdana,sans-serif')"`
TITLEFONTSIZE	Font size of title text. Value: Size with unit, in single quotes. Unit ('px', 'pt', 'em' etc.) is mandatory. If the value is an empty string `''`, the fontsize of the tooltip body is applied.
WIDTH	Width of tooltip. Value: Integer ≥ 0. If 0 (the default), the width is automatically adapted to the content of the tooltip. Note that the tooltips follow the W3C box model, which means that the specified width applies to the actual content of the tooltip, excluding padding (see PADDING command), border and shadow.

Fichiers source prétraités

Cette fonctionnalité n'est pas disponible pour Createxo. Vous pouvez l'utiliser si vous écrivez des exercices OEF (et Deductio) sous Modtool.

Elle vous permet d'inclure des parties de codes communes dans plusieurs fichiers OEF. Vous pouvez également l'utiliser pour fabriquer par lots plusieurs oef à partir d'un pré-source, chacun différant des autres par quelques définitions de macros.

Pour utiliser cette possibilité, un sous-répertoire cpp/ doit être créée sous src. Y mettre les fichiers de pré-source d'extension .cpp. Dans ces fichiers, vous pouvez ajouter des directives cpp comme `#include`, `#define` ou `#if`. (Veuillez vous référer aux manuels cpp et aux spécifications c pour les détails de la syntaxe.)

La première ligne du fichier cpp doit définir une liste de cibles (targets), dans le format

```
target=targ1 targ2 targ3 ...
```

Cette ligne indique que ce fichier cpp doit fabriquer targ1.oef, targ2.oef, etc (l'extension .oef sera donc ajoutée aux noms de sortie). Lors de la génération du fichier cible *targ1.oef*, une macro *TARGET_targ1* est définie. Par conséquent, dans le source cpp, vous pouvez ajouter des lignes comme

```
#if defined TARGET_targ1
 \title{Exercise 1}
 \integer{v1=3}
 \text{t1=this is target 1}
#endif
#if defined TARGET_targ2
 \title{Exercise 2}
 \integer{v1=5}
 \text{t1=this is target 2}
#endif
```

afin de rendre le contenu dépendant de la cible.

Le nom du fichier (dans l'exemple `targ1`, ...) est accessible par la variable `OEFFILE`.

Dans l'exemple suivant, le titre de l'exercice sera `targ1`

```
#if defined TARGET_targ1
 \title{OEFFILE}
 \integer{v1=3}
 \text{t1=dit is target 1}
#endif
```

Les fichiers `include` doivent également être placés dans le répertoire src/cpp, avec extension .inc. Une ligne

```
#include "common.inc"
```

va insérer le contenu de common.inc dans le fichier généré OEF. Veuillez noter que pour des raisons de sécurité, il est interdit de spécifier des répertoires de fichiers include.

Une remarque spéciale: cpp s'embrouille lorsque votre fichier contient dans le texte des apostrophes ou guillemets qui ne sont pas fermés. Dans ce cas, vous pouvez protéger les commandes contenant ces textes par des commentaires C (`/* ... */`).

> Attention : Dans les commandes commençant par # comme dans `#include` , le caractère `#` doit être le premier caractère de la ligne. Il ne faut pas non plus utiliser `#` comme caractère de commentaires.

Environnement dans un exercice

La variable spéciale `\oefenv` est utilisée pour tester un exercice (si elle n'est pas redéfinie dans l'exercice).

Pour l'instant, le seul contenu possible de cette variable est le mot ``debug''. Ce mot apparaît à l'intérieur de `\oefenv` dans les situations suivantes.

1. quand l'exercice est exécuté à l'intérieur de Createxo ;
2. quand il est testé dans Modtool par le développeur lui-même
3. quand l'exercice est dans une classe et exécuté par l'administrateur de la classe.

Dans ces cas, la bonne réponse apparaît par défaut dans les champs de réponse si vous le demandez, vous permettant de voir ce que l'exercice attend comme réponse sans avoir besoin de remplir vous-même les champs.

Vous pouvez ajouter des informations de débogage à l'intérieur de votre exercice en utilisant des conditions comme

```
\if{debug iswordof \oefenv}{informations de débogage à ajouter ici}
```

Ces informations sont alors automatiquement montrées quand l'exercice est sous test et automatiquement cachées quand les étudiants y travaillent.

Toutes les comparaisons sont faites sur des chaînes de caractères : `string1 rel string2` . Plusieurs comparaisons peuvent être reliées par les opérateurs `and` et `or` . Les parenthèses sont alors utilisées pour construire des expressions logiques complexes de comparaison.

Relations

Relation	Exemple	Explications
`==`	`string1 == string2` `string1 = string2`	avec `if` : vrai si `string1` et `string2` sont identiques ; avec `ifval` : vrai si les évaluations numériques de `string1` et de `string2` sont égales.
`!=`	`string1 != string2` `string1 <> string2`	avec `if` : vrai si `string1` et `string2` ne sont pas identiques ; avec `ifval` :vrai si les évaluations numériques de `string1` et de `string2` ne sont pas égales ;
`<`	`string1 < string2`	vrai si l'évaluation numérique de `string1` est strictement inférieure à celle de `string2`
`<=`	`string1 <= string2`	vrai si l'évaluation numérique de `string1` est inférieure ou égale à celle de `string2` .
`>`	`string1 > string2`	vrai si l'évaluation numérique de `string1` est strictement supérieure à celle de `string2` .
`>=`	`string1 >= string2`	vrai si l'évaluation numérique de `string1` est supérieure à celle `string2` .
`isin`	`string1 isin string2`	vrai si `string1` est une sous-chaîne de caractères de `string2` .
`notin`	`string1 notin string2`	vrai si `string1` n'est pas une sous-chaîne de caractères de `string2` .
`iswordof`	`string1 iswordof string2`	vrai si `string1` est un mot de `string2` .
`notwordof`	`string1 notwordof string2`	vrai si `string1` n'est pas un mot de `string2` .
`isvarof`	`string1 isvarof string2`	vrai si `string1` est une variable mathématique de l'expression `string2` .
`notvarof`	`string1 notvarof string2`	si `string1` n'est pas une variable mathématique de l'expression `string2` .
`isvariableof`	`string1 isvariableof string2`	vrai si `string1` est une variable mathématique de l'expression `string2` .
`notvariableof`	`string1 notvariableof string2`	vrai si `string1` n'est pas une variable mathématique de l'expression `string2` .
`isitemof`	`string1 isitemof string2`	vrai si `string1` est un item de la liste `string2` .
`notitemof`	`string1 notitemof string2`	vrai si `string1` n'est pas un item de la liste `string2` .
`islineof`	`string1 islineof string2`	vrai si `string1` est une ligne de `string2` .
`notlineof`	`string1 notlineof string2`	vrai si `string1` n'est pas une ligne de `string2` .
`issamecase`	`string1 issamecase string2`	vrai si `string1` et `string2` sont les mêmes textes à des espaces multiples près, mais tenant compte de la casse des lettres.
`notsamecase`	`string1 notsamecase string2`	vrai si `string1` et `string2` ne vérifient pas le critère ci-dessus.
`issametext`	`string1 issametext string2`	vrai si `string1` et `string2` sont les mêmes textes à des espaces multiples près, à la casse près et aux lettres accentuées près.
`notsametext`	`string1 notsametext string2`	vrai si `string1` et `string2` ne vérifient pas le critère précédent.

Bibliothèque de scripts (slib)

Les scripts de cette bibliothèque peuvent être appelés d'un module en utilisant la commande `!read` (ou `!readproc` à partir d'un fichier phtml). Par exemple, la ligne

```
!read slib/matrix/random 3, 5, 10
```

permet de construire une matrice 3×5 avec des coefficients entiers aléatoires dans [-10, 10]. Le résultat est affecté à la variable `slib_out` . Pour appeler un script slib d'un exercice OEF, d'un document ou dans un message du forum, on utilise la fonction `slib()` . Par exemple `\text{a=slib( ... ) }`

Seules les variables préfixées par `slib_` sont modifiées par ces scripts.

Liste de scripts disponibles

Tous (274)
algebra (4)
analysis (6)
chemistry (21)
circuits (6)
data (4)
draw (16)
function (3)
games (3)
geo2D (5)
geo3D (10)
graph (13)
graphpaper (8)
lang (12)
life (1)
list (1)
matrix (13)
media (5)
numeration (6)
oef (9)
polynomial (1)
set (1)
stat (99)
text (20)
triplerelation (1)
utilities (6)

Tous

Nom	Résultat
algebra/partitionconj	Partition conjuguée [Détail]
algebra/partitiondraw	Dessiner des diagrammes de Young d'une partition [Détail]
algebra/partitionlex	Nouvelle partition pour l'ordre lexicographique décroissant [Détail]
algebra/slopedraw	Polygone tracé à partir des pentes (dessin) [Détail]
analysis/inversedomain	Image réciproque de régions [Détail]
analysis/odejs	Tracés de solutions d'un système différential (avec jsxgraph). [Détail]
analysis/odejs2	Dessin de solutions d'un système différentiel, dans le plan (t, x/y) et dans le plan (x,y). [Détail]
analysis/odephase	Portrait de phase d'un système différentiel autonome [Détail]
analysis/rungekutta	Equation différentielle (par Runge-Kutta) experimental en faire une liste sans dessin [Détail]
analysis/slopefield	Champ de direction (par exemple pour un système différentiel) [Détail]
chemistry/brut2html	forme HTML de la formule d'une molécule [Détail]
chemistry/chemeq_add	Calcule une combinaison d'équations chimiques [Détail]
chemistry/chemeq_compare	Compare des équations chimiques [Détail]
chemistry/chemeq_components	Composants chimiques [Détail]
chemistry/chemeq_el	Retourne le nombre d'électrons dans une réaction d'oxydoréduction [Détail]
chemistry/chemeq_equilibrium	Analyse de l'équilibre dans les formules chimiques [Détail]
chemistry/chemeq_mass	Masse molaire [Détail]
chemistry/chemeq_rev	Renvoie une équation chimique inversée [Détail]
chemistry/chemeq_rq	Quotients de réaction composée et lois de Nernst pour les équations chimiques [Détail]
chemistry/chemeq_tex	Composés moléculaire et équations chimiques [Détail]
chemistry/chemshow	Dessin d'une molécule en 2D [Détail]
chemistry/cram	Représentation de Cram [Détail]
chemistry/jmolbutton	Bouton dans Jmol. Doit apparaitre APRES l'applet Jmol. [Détail]
chemistry/jmolcheckbox	Bouton Checkbox (bouton carré de validation) dans Jmol [Détail]
chemistry/jmolradiogroup	Radiobuttons (boutons ronds) dans Jmol [Détail]
chemistry/jmolshow	Applet Jmol [Détail]
chemistry/leftind	Indices et exposants à gauche et droite [Détail]
chemistry/molarmass	Masse molaire [Détail]
chemistry/molecule	Tableau périodique [Détail]
chemistry/moleculeViewer	Visualiseur de molécules sous Java [Détail]
chemistry/newman	Projection de Newman [Détail]
circuits/complist	List available circuit components [Détail]
circuits/comppos	Component position information of a circuit type. [Détail]
circuits/draw	Draw circuit scheme according a circuit type. [Détail]
circuits/drawcomp	Draw circuit components according to a circuit type. [Détail]
circuits/drawwire	Draw the fixed circuit wiring of a given circuit type. [Détail]
circuits/range	Size and range information of a circuit type. [Détail]
data/columnsort	Sort data according to a column [Détail]
data/randline	Take a random line of a data file [Détail]
data/random	Randomly selects a number of (different) objects [Détail]
data/randrec	Take a random field of a record file [Détail]
draw/balance	Balance (Roberval) [Détail]

draw/brokenlinegraph	Trace une fonction affine par morceaux continue en connaissant les points de changement de pente [Détail]
draw/clock	Dessine une horloge réglée sur l'heure donnée [Détail]
draw/convpixel	Conversion de coordonnées [pixels] <-> [repère mathématique] [Détail]
draw/domino	Domino [Détail]
draw/drtgraduee	Droite graduée [Détail]
draw/graphviz	Graphviz [Détail]
draw/graphvizpoints	Coordonnées des noeuds d'un graphe créé précédemment avec graphviz. [Détail]
draw/meter	Compteur avec aiguille configurable. [Détail]
draw/polygon	Polygone régulier [Détail]
draw/radar	Radar [Détail]
draw/randpolygon	Polygone quelconque [Détail]
draw/range	Produit d'intervalles [Détail]
draw/repdroite	Calcule les coordonnées des deux points extrêmes pour tracer une droite dans un repère [Détail]
draw/repere	Trace un repère [Détail]
draw/thermometer	Thermomètre avec niveau configurable [Détail]
function/bounds	Les bornes d'une fct réelle à une variable sur un intervalle [x1,x2] [Détail]
function/bounds2	Des bornes d'une fonction réelle à deux variables dans un rectangle [x1,x2], [y1,y2] [Détail]
function/integrate	Intégration définie ou non d'une fonction à une variable [Détail]
games/chessboard	Echiquier [Détail]
games/chessimage	Echiquier (image) [Détail]
games/chessmv	Mouvement dans un jeu d'échecs [Détail]
geo2D/geogebra	Applet GeoGebra (html5) [Détail]
geo2D/geogebra3	Applet GeoGebra (java) [Détail]
geo2D/ggb2jsxgraph	Geogebra to Jsxgraph [Détail]
geo2D/jsxgraph	Plugin pour JSXGraph [Détail]
geo2D/squaretile	Réseau de carrés [Détail]
geo3D/3Dviewer	Visualiseur 3D [Détail]
geo3D/CaR	applet 3D avec C.a.R [Détail]
geo3D/Convex3D	Applet polyèdre avec Convex3D [Détail]
geo3D/draw	Polyèdre avec flydraw [Détail]
geo3D/drawtile	Réseaux de cubes [Détail]
geo3D/off2jmol	Conversion du format off en un script jmol [Détail]
geo3D/off2xyz	Conversion du format off au format xyz [Détail]
geo3D/polyhedra	Applet de tracé de polyèdre avec C.a.R [Détail]
geo3D/polyhedradual	Applet de tracé de polyèdre et de son dual avec C.a.R [Détail]
geo3D/threeD	Applet ThreeD [Détail]
graph/connexcomponent	Composantes connexes d'un sommet dans un graphe [Détail]
graph/connexity	Composantes connexes d'un graphe simple [Détail]
graph/distance	Matrice du diamètre d'un graphe [Détail]
graph/draw	Dessin de graphe [Détail]
graph/drawcc	dessin de graphe avec une composante connexe colorée [Détail]
graph/drawtree	Dessin d'arbre [Détail]
graph/gpt	graphe orienté sans circuit [Détail]
graph/graphviz	Graphviz [Détail]
graph/path	Chemins dans un graphe [Détail]
graph/randomconnex	Graphe connexe aléatoire [Détail]
graph/randomeuler	Graphe eulérien aléatoire [Détail]
graph/randtree	Arbre aléatoire [Détail]
graph/shortpath	Plus court chemin dans un graphe [Détail]
graphpaper/correct_milli	Graphic paper sheet with red correct plot preloaded [Détail]
graphpaper/func	One function plot, ready to append to a previously made graph paper [Détail]

graphpaper/func_milli	Graphic paper sheet with function plot and red correct plot preloaded [Détail]
graphpaper/imgpoints	Utility for a clickable graphic paper sheet [Détail]
graphpaper/millimetre	Graphic paper sheet [Détail]
graphpaper/strings	Prepare strings to be written on a graphic paper sheet [Détail]
graphpaper/tograph	Utility for a clickable graphic paper sheet [Détail]
graphpaper/whereclick	Utility for a clickable graphic paper sheet [Détail]
lang/enword2ipa	IPA transcription of english words [Détail]
lang/epd2ipa	IPA transcription according to epd ascii codage (for english) [Détail]
lang/fname	Prénoms au hasard [Détail]
lang/fraccord	Accord of French adjectives and names [Détail]
lang/frapostrophe	Apostrophe reduction of a French text [Détail]
lang/frartdef	Transform a French noun into definite form [Détail]
lang/frcodcoi	Find a random complement of a French verb [Détail]
lang/frverbconj	The conjugation of a French verb [Détail]
lang/images	Images in some datamodule [Détail]
lang/randomword	Output random words in the dictionary [Détail]
lang/sampa2ipa	IPA transcription according to Sampa Ascii codage (for english) [Détail]
lang/swac	Insertion of words (audio) from the swac packs [Détail]
life/frcommodity	Give a random commodity with given price, French [Détail]
list/selshuf	Selective shuffle [Détail]
matrix/concate	concatenation [Détail]
matrix/det	The determinant of a square matrix [Détail]
matrix/givenrank	Generates a random matrix of given rank [Détail]
matrix/inverse	The inverse of a square matrix [Détail]
matrix/invertible	Generates a random invertible matrix [Détail]
matrix/itriangular	Generates a random invertible triangular matrix [Détail]
matrix/non0	Matrice au hasard ayant des coefficients non nuls [Détail]
matrix/orthogonal	Generates a random orthogonal matrix [Détail]
matrix/random	Generates a random matrix [Détail]
matrix/trace	The trace of a square matrix [Détail]
matrix/transpose	The transpose of a matrix [Détail]
matrix/triangular	Generates a random triangular matrix [Détail]
matrix/unimodular	Generates a random unimodular matrix [Détail]
media/audio	Audio insertion [Détail]
media/dewplayer	Audio insertion with dewplayer [Détail]
media/player	Audio insertion with hbs_mp3_player [Détail]
media/player_mp3_multi	Audio insertion with player_mp3_multi [Détail]
media/video	Insertion d'une vidéo [Détail]
numeration/babylonien	Ecriture Babylonienne d'un nombre entier [Détail]
numeration/baseblock	Blocs de base en numération [Détail]
numeration/basep	Passage de la base dix vers la base p. [Détail]
numeration/ecriturelettre	Écriture d'un nombre en lettres. [Détail]
numeration/ecriturenombre	Ecriture d'un nombre avec regroupement des chiffres par trois. [Détail]
numeration/egyptien	Ecriture égyptienne d'un nombre entier [Détail]
oef/blank	Gestion des blancs [Détail]
oef/codelim	OEF code length limit register [Détail]
oef/codename	Register OEF code reply name allow/deny [Détail]
oef/env	Get an OEF environment variable [Détail]
oef/insfilename	Output the file name of the last insert [Détail]
oef/newfile	Save a text in a file [Détail]
oef/postsrc	OEF code input postpender [Détail]
oef/presrc	OEF code input prepender [Détail]
oef/sortorder	Sort order [Détail]

polynomial/random	Random polynomial [Détail]
set/subset	Sous-ensembles d'un ensemble [Détail]
stat/1d	Computes 1-dimensional statistical data [Détail]
stat/arithmean	Arithmetic mean of statistical data [Détail]
stat/beta	Simulation de réalisations d'une variable aléatoire suivant la loi beta [Détail]
stat/betacdf	Fonction de répartition de la loi Beta [Détail]
stat/betainv	Quantile de la loi Beta [Détail]
stat/betapdf	Densité de probabilité de la loi Beta [Détail]
stat/binomial	Simulation de réalisations d'une variable aléatoire suivant la loi binomiale. [Détail]
stat/binomialcdf	Fonction de répartition de la loi binomiale [Détail]
stat/binomialinv	Quantile de la loi binomiale [Détail]
stat/binomialpdf	Densité de probabilité de la loi binomiale [Détail]
stat/boxplot	Box plot [Détail]
stat/cauchy	Simulation de réalisations d'une variable aléatoire suivant la loi de Cauchy. [Détail]
stat/cauchycdf	Fonction de répartition de la loi de Cauchy [Détail]
stat/cauchyinv	Quantile de la loi de Cauchy [Détail]
stat/cauchypdf	Densité de probabilité de la loi de Cauchy [Détail]
stat/chi2	Simulation de réalisations d'une variable aléatoire suivant la loi du chi-deux [Détail]
stat/chi2cdf	Fonction de répartition de la loi du chi-deux [Détail]
stat/chi2inv	Quantile de la loi du chi-deux [Détail]
stat/chi2pdf	Densité de probabilité de la loi du chi-deux [Détail]
stat/correlation	Matrix of correlation [Détail]
stat/covariance	Matrix of covariance [Détail]
stat/deviation	Deviation of statistical data [Détail]
stat/discretelaw	Generation of a discrete law with nonnegative coefficients [Détail]
stat/effectif	Effectifs of statistical series in classes [Détail]
stat/empiric	Simulation de réalisations d'une variable aléatoire suivant la loi discrète [Détail]
stat/exponential	Simulation de réalisations d'une variable aléatoire suivant la loi exponentielle. [Détail]
stat/exponentialcdf	Fonction de répartition de la loi exponentielle [Détail]
stat/exponentialinv	Quantile de la loi exponentielle [Détail]
stat/exponentialpdf	Densité de probabilité de la loi exponentielle [Détail]
stat/fisher	Simulation de réalisations d'une variable aléatoire suivant la loi de Fisher. [Détail]
stat/fishercdf	Fonction de répartition de la loi de Fisher [Détail]
stat/fisherinv	Quantile de la loi de Fisher [Détail]
stat/fisherpdf	Densité de probabilité de la loi de Fisher [Détail]
stat/freq	Frequencies of statistical data [Détail]
stat/gamma	Simulation de réalisations d'une variable aléatoire suivant la loi Gamma. [Détail]
stat/gammacdf	Fonction de répartition de la loi Gamma [Détail]
stat/gammainv	Quantile de la loi Gamma [Détail]
stat/gammapdf	Densité de probabilité de la loi Gamma [Détail]
stat/geomean	Geometric mean of data [Détail]
stat/geometric	Simulation de réalisations d'une variable aléatoire suivant la loi géométrique sur N [Détail]
stat/geometric1	Simulation de réalisations d'une variable aléatoire suivant la loi géométrique sur N* [Détail]
stat/geometric1cdf	Fonction de répartition de la loi géometrique sur N* [Détail]
stat/geometric1inv	Quantile de la loi géométrique sur N* [Détail]
stat/geometric1pdf	Densité de probabilité de la loi géometrique sur N* [Détail]
stat/geometriccdf	Fonction de répartition de la loi géométrique sur N [Détail]
stat/geometricinv	Quantile de la loi géométrique sur N [Détail]
stat/geometricpdf	Densité de probabilité de la loi géometrique sur N [Détail]

stat/harmonic	Harmonic mean of statistical data [Détail]
stat/histo	Histogram [Détail]
stat/hypergeometric	Simulation de réalisations d'une variable aléatoire suivant la loi hypergéométrique [Détail]
stat/hypergeometriccdf	Fonction de répartition de la loi hypergéométrique [Détail]
stat/hypergeometricinv	Quantile de la loi hypergéométrique [Détail]
stat/hypergeometricpdf	Densité de probabilité de la loi hypergéométrique [Détail]
stat/laplace	Simulation de réalisations d'une variable aléatoire suivant la loi de Laplace [Détail]
stat/laplacecdf	Fonction de répartition de la loi de Laplace [Détail]
stat/laplaceinv	Quantile de la loi de Laplace [Détail]
stat/laplacepdf	Densité de probabilité de la loi de Laplace [Détail]
stat/linearcong	Generation of linear congruential random integers [Détail]
stat/logistic	Simulation de réalisations d'une variable aléatoire suivant la loi logistique [Détail]
stat/logisticcdf	Fonction de répartition de la loi logistique [Détail]
stat/logisticinv	Quantile de la loi logistique [Détail]
stat/logisticpdf	Densité de probabilité de la loi logistique [Détail]
stat/lognormal	Simulation de réalisations d'une variable aléatoire de loi lognormale [Détail]
stat/lognormalcdf	Fonction de répartition de la loi log-normale [Détail]
stat/lognormalinv	Quantile de la loi log-normale [Détail]
stat/lognormalpdf	Densité de probabilité de la loi log-normale [Détail]
stat/median	Data median [Détail]
stat/multinomial	Simulation de réalisations d'une variable aléatoire suivant la loi multinomiale [Détail]
stat/nbin	Simulation de réalisations d'une variable aléatoire suivant la loi binomiale négative [Détail]
stat/nbincdf	Fonction de répartition de la loi binomiale négative. [Détail]
stat/nbininv	Quantile de la loi binomiale négative [Détail]
stat/nbinpdf	Densité de probabilité de la loi binomiale négative. [Détail]
stat/normal	Simulation de réalisations d'une variable aléatoire suivant la loi gaussienne [Détail]
stat/normalcdf	Fonction de répartition de la loi normale [Détail]
stat/normalinv	Quantile de la loi normale [Détail]
stat/normalpdf	Densité de probabilité de la loi normale [Détail]
stat/pascal	Simulation de réalisations d'une variable aléatoire suivant la loi de Pascal [Détail]
stat/pascalcdf	Fonction de répartition de la loi de Pascal [Détail]
stat/pascalinv	Quantile de la loi de Pascal [Détail]
stat/pascalpdf	Densité de probabilité de la loi de Pascal [Détail]
stat/poisson	Simulation de réalisations d'une variable aléatoire suivant la loi de Poisson. [Détail]
stat/poissoncdf	Fonction de répartition de la loi de Poisson [Détail]
stat/poissoninv	Quantile de la loi de Poisson [Détail]
stat/poissonpdf	Densité de probabilité de la loi de Poisson [Détail]
stat/posdiscretelaw	Generation of a discrete law with positive coefficients [Détail]
stat/prod	Product of data [Détail]
stat/quadratic	Quadratic mean [Détail]
stat/random	Generation of random numbers [Détail]
stat/range	Data range [Détail]
stat/student	Simulation de réalisations d'une variable aléatoire suivant une loi de Student. [Détail]
stat/studentcdf	Fonction de répartition de la loi de Student [Détail]
stat/studentinv	Quantile de la loi de Student [Détail]
stat/studentpdf	Densité de probabilité de la loi de Student [Détail]
stat/sum	Data sum [Détail]
stat/variance	Variance [Détail]
stat/weibull	Simulation de réalisations d'une variable aléatoire suivant la loi de Weibull. [Détail]

stat/weibullcdf	Fonction de répartition de la loi de Weibull [Détail]
stat/weibullinv	Quantile de la loi de Weibull [Détail]
stat/weibullpdf	Densité de probabilité de la loi de Weibull [Détail]
text/approximation	Calcul d'intervalle approché pour un réel donné [Détail]
text/balloon	Bulles de texte (façon cartoon) [Détail]
text/cdecomment	Extract comment from a c source code. [Détail]
text/comblin	Simplify a linear combination [Détail]
text/crossword	Crossword [Détail]
text/cutchoice2	Cut out embedded choices for OEF [Détail]
text/cutchoices	Cut out embedded choices for OEF [Détail]
text/markerror	For marking words with mistake [Détail]
text/markgroup	For marking group of words with given explanation [Détail]
text/marktext	Texte for use with type mark for OEF (word) [Détail]
text/marktextpartial	For marking some words with given explanation [Détail]
text/markword	For use with type mark in OEF [Détail]
text/matrixhtml	Transform a matrix into html matrix (table). [Détail]
text/matrixinsert	Insert a coefficient in a matrix [Détail]
text/matrixtex	Matrix in Latex [Détail]
text/maximamatrix	Transform a matrix to maxima format [Détail]
text/octavematrix	Transform an octave output matrix into standard format [Détail]
text/sigunits	Make a representation of a physical quantity with a given number of significative digits [Détail]
text/spirale	Write on a spirale [Détail]
text/whitespace	Replace white spaces [Détail]
triplerelation/tabular	Double entry table for training to relations between three quantities [Détail]
utilities/date	Date [Détail]
utilities/mathcalc	Mathcalc [Détail]
utilities/nopaste	No copy-paste [Détail]
utilities/notepad	Notepad [Détail]
utilities/tooltip	Tooltip containing an html text which appears when the mouse points on a word. [Détail]
utilities/trigo-calc	Inline Trigonometric calculator [Détail]

Instructions

instruction	nombre de paramètres	description	paramètres optionnels	mots d'option reconnus
title	1	définit le titre de l'exercice		
language	1	définit la langue de l'exercice, comme en ou fr		
author	1	définit l'auteur de l'exercice. Le mettre sous la forme Prénom, Nom (dans le cas de plusieurs auteurs, les séparer par des points-virgules).		
email	1	définit l'adresse électronique de l'auteur (dans le cas de plusieurs auteurs, les séparer par une virgule).		
format	1	format de l'énoncé		
css	1	définit le style css		
keywords	1	Mots clés de l'exercice (prendre de préférence les mots clés officiels séparés par des virgules)		
credits	1	permet d'inclure automatiquement un remerciement ou un crédit en fin d'exercice (les variables sont évaluées).		
description	1	description de l'exercice destinée à l'élève		
observation	1	description de l'exercice destinée à l'enseignant		
precision	1	précision en comparant la réponse de l'utilisateur avec la solution. Donnez un nombre positif n ici : la comparaison sera effectuée avec une tolérance de 1/n.		
range	1	zone de variables pour l'évaluation de fonction fournie par l'utilisateur. Doit être donnée sous forme n1..n2, où n1 est le point de départ, n2 le point d'arrivée.		
computeanswer	1	La commande \computeanswer{ no } précise que l'utilisateur doit lui-même faire les calculs et entrer la valeur finale. Si par contre, on met \computeanswer{ yes }, l'utilisateur peut entrer une formule comme 5*5, laissant à l'ordinateur le soin de faire les calculs.		

statement	1	le paramètre est l'énoncé de l'exercice		
answer	2-5	définit une réponse libre. Le premier paramètre est le message pour la réponse, et le second est la bonne réponse. La réponse sera analysée selon des types (nombre, fonction, texte, etc).	type option weight	reorder shuffle nonstop
choice	3-5	définit un choix multiple. Le premier paramètre est le message pour le choix, le second les bons choix, et le troisième les mauvais choix. Les deux derniers paramètres peuvent (doivent) être une liste d'objets séparés par des virgules. Il est permis d'avoir plusieurs bons choix. Si un choix apparait à la fois comme bon et mauvais, il est pris pour bon.	option weight	shuffle noidontknow
condition	2-4	définit une condition spéciale pour l'évaluation de réponses libres. Le premier argument est un texte qui sera affiché lors de l'analyse de la réponse. Le second argument, on met la liste des conditions que la réponse de l'utilisateur doit satisfaire pour être considérée comme bonne.	option weight	hide
solution	1	donne une solution expliquée de l'exercice. Le gestionnaire OEF peut décider de montrer la solution ou pas à l'utilisateur, suivant le choix du niveau de difficulté pris par l'utilisateur. Ne peut apparaître qu'une seule fois dans un exercice.		
hint	1	donne une indication de l'exercice. Le gestionnaire OEF peut décider de montrer l'indication ou non, suivant le niveau de difficulté. Ne peut apparaître qu'une seule fois dans un exercice.		
help	1	donne une aide à l'exercice. Cette aide sera toujours accessible à l'utilisateur, dans une fenêtre `popup'. Ne peut apparaître qu'une seule fois dans un exercice.		
feedback	2	affiche un commentaire quand la réponse tombe sous une certaine condition. Peut normalement être utilisé pour avertir d'une erreur typique.		
steps	1	sert à définir les questions apparaissant à chaque étape ; doit être mis avant la commande `statement` (voir la variable `\step`). Ne peut apparaître qu'une seule fois dans un exercice.		
nextstep	1	sert à définir de manière dynamique les questions qui devront être posées ; doit être mis avant la commande `statement` (voir la variable `\step`). Can be used only one time in the exercise.		
conditions	1	permet d'indiquer les numéros des conditions utiles pour l'exercice servant à contrôler les réponses de l'utilisateur.		
latex	1	permet d'écrire une version en latex de l'exercice utilisant les variables définies dans l'exercice et pouvant être téléchargées dans la version imprimable de l'exercice (accessible uniquement par les développeurs ou les enseignants d'une classe). Il est conseillé de mettre l'énoncé dans un environnement latex prédéfini `\begin{statement} \end{statement}` et la solution dans l'environnement `\begin{sol} ... \end{sol}`.		

CSS

text-left	align to the left
center	center
wimscenter	center with margins
blockcenter	center a block as a table
white	background in white
float_right	text float on the right
float_left	text float on the left
clearall	use it to stop float for example with br/
clearfix	use it to contain floats without altering what's next.
small	text is smaller
smaller	text smaller than small
bold	text in bold
larger	text larger
wims_emph	emphasis some part of the text
inline	text in line (if you put inline directly on li, there will be no margin)
inline	Inline general lists
inline	Inline Ordered Lists
inline	Inline Unordered Lists
wims_nopuce	list without puce
spacer	use spacer on div to add extra spaces

spacer	use spacer on lists (ul/ol) to add extra spaces on all li
wims_audio	default for audio
oef_indgood	emphasing results of the student: good answer
oef_condgood	emphasing results of the student: good condition
oef_indbad	emphasing results of the student: bad answer
oef_condbad	emphasing results of the student: bad condition
oef_indforget	emphasing results of the student: forgotten item in the answer
oef_indpartial	emphasing results of the student: partial answer
oef_indprec	emphasing results of the student: bad precision
jxgbox	style for including a jsxgraph figure
wims_contribute	contributor citation or credits
wims_credits	contributor citation or credits
wims_instruction	Technical instructions in exercises for example
wims_difficultylevel	Difficulty level of the exercise or any other hint on the level
wims_smallhelp	small help
wims_smallremark	remark in small