



# Developpez

*Le Mag*

Édition de juin - juillet 2013.

Numéro 46.

Magazine en ligne gratuit.

Diffusion de copies conformes à l'original autorisée.

Réalisation : Alexandre Pottiez

Rédaction : la rédaction de Developpez

Contact : [magazine@redaction-developpez.com](mailto:magazine@redaction-developpez.com)

## Sommaire

Réseau	Page 2
Mac	Page 8
PHP	Page 17
Developpement Web	Page 23
Dév. Web Java	Page 26
Java	Page 33
Android	Page 40
Perl	Page 42
OpenOffice/LibreOffice	Page 46
2D/3D/Jeux	Page 55
Liens	Page 65

## Article Réseau



### L'internet Rapide et Permanent

Parlons de la fibre optique et des technologies sous-jacentes.

par **Christian Caleca**

Page 2



## Article OpenOffice/LibreOffice

## Éditorial

Retrouvez le meilleur cahier de vacances pour cet été. Il est riche en développements et nouvelles technologies.

Profitez-en bien !

La rédaction

### Présentation de LibreOffice

Découvrez la célèbre suite bureautique.

par **Sophie Gautier**

Page 46

### L'Internet Rapide et Permanent

Vous disposez d'une connexion permanente et rapide... et maintenant, vous êtes perdu dans la technique...

Cette série « L'Internet Rapide et Permanent », que Christian Caleca nous a aimablement autorisés à reproduire, est là pour répondre à quelques-unes de ces questions. Cet article parlera de la fibre optique et des technologies sous-jacentes.

#### 1. Introduction



Elle n'a jamais autant été à la mode, mais qu'est-ce exactement ? Et à quoi sert-elle ? Pour aller vite, c'est un fil de verre, entouré d'une gaine « réfléchissante ». Sa propriété principale est de servir de « tuyau » dans lequel on peut faire circuler de la lumière. En plus de servir à construire tout un tas de gadgets amusants, on peut lui trouver quelques applications plus technologiques, allant de l'endoscopie au transfert de données numériques. Vous l'aurez deviné, c'est plutôt cet aspect-là qui va nous intéresser.

Bien entendu, dans ce domaine d'application, la fibre optique se met à ressembler furieusement à n'importe quel câble électrique et perd beaucoup de son aspect poétique. (C'est également le cas pour l'endoscope.)

Pourquoi donc essayer de transporter de l'information numérique de cette manière ? Qu'est-ce qu'on y gagne ?

Autant de questions qui trouveront, je l'espère, des réponses dans les pages qui suivent. Je vous rassure tout de suite, il n'est pas question de se lancer dans de brillantes démonstrations appuyées par de vastes calculs, mais simplement d'essayer d'expliquer le principe.



#### 2. Éclairons notre lanterne

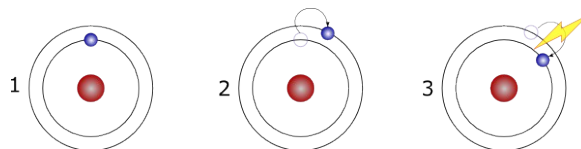
La théorie de la lumière est une théorie particulièrement obscure. Nous allons passer beaucoup de temps à utiliser de faux modèles pour expliquer des phénomènes « vrais », dans la mesure où l'on peut les vérifier par la pratique.

#### 2.1. Onde ou Photon ?

Pour expliquer certaines observations, il faut que la lumière soit une onde. Pour en expliquer d'autres, il faut qu'elle soit un flux de particules. Qu'à cela ne tienne, nous nous en sortirons quand même, la mauvaise foi n'ayant jamais étouffé un scientifique. Louis de Broglie (Fr., 1892-1987) a avancé en 1924 que les corpuscules de matière étaient accompagnés d'une onde, ce qui, en quelque sorte, résout le problème de façon assez élégante.

#### 2.2. Qu'est-ce que la lumière ?

Nous allons utiliser un modèle d'atome (faux), celui de Bohr. En fait, il n'est pas tout à fait faux, mais il n'est pas juste non plus ; ça ne fait rien, il permet d'expliquer l'émission de la lumière d'une manière tout à fait acceptable.



- Bohr explique que, dans un atome, les électrons tournent autour du noyau selon des orbites bien définies. Ce n'est pas tout à fait juste, mais presque. C'est en tout cas suffisant comme précision pour notre propos.
- Lorsque l'on excite les électrons, par exemple en les chauffant, ces électrons récupèrent de l'énergie. Ce surcroît d'énergie les fait passer sur une orbite supérieure.
- Comme la situation est instable, ils finissent par revenir sur leur orbite « normale » en restituant l'énergie qu'ils avaient empruntée. Cette restitution d'énergie se fait sous la forme d'émission de lumière.

On dit que l'électron franchit des niveaux d'énergie. C'est grâce à cette particularité de la physique atomique que Thomas Edison est devenu célèbre. Dans une ampoule électrique, les atomes du filament, chauffés par effet Joule au passage du courant électrique, montent des niveaux d'énergie supérieurs et émettent de la lumière chaque fois qu'ils redescendent sur un niveau inférieur.

Le problème, en ces temps d'écologie, d'énergie non nucléaire, mais renouvelable, de réchauffement atmosphérique, c'est que la quantité d'énergie fournie pour

chauffer le filament est énormément plus importante que l'énergie lumineuse récupérée. Le rendement est de l'ordre de 2 % en moyenne pour une ampoule incandescente à filament de tungstène. D'où l'idée « lumineuse » d'en interdire la vente à partir de 2010. La bougie alors ? 0,04 % de rendement... Le tube fluorescent ? Autour de 15 % de rendement. C'est nettement mieux, mais c'est aussi nettement plus polluant chimiquement et électromagnétiquement parlant. Le meilleur rendement est actuellement obtenu avec de la vapeur de sodium en basse pression : 27 %. Certes, c'est un peu orangé comme lumière. Le mieux serait de se passer de lumière ; ça se faisait très bien à l'époque des cavernes. Mais ne sortons pas du sujet...

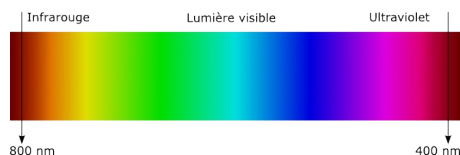
### 2.3. Blanche ou colorée ?

Lorsqu'un électron redescend sur une couche inférieure, il émet une lumière monochromatique. La couleur dépend du niveau d'énergie descendu. La lumière apparaît blanche parce qu'il y a beaucoup de niveaux d'énergie différents mis en œuvre et qu'il y a donc beaucoup de radiations de couleurs différentes qui sont émises. Leur somme donne une lumière blanche. Nous sommes ici en synthèse additive et la somme de toutes les couleurs donne du blanc.

Dans le cas de corps simples comme le néon, il n'y a que deux niveaux d'énergie et donc une seule couleur, dans l'orange.

Avec l'argon aussi, il n'y a que deux niveaux. Manque de chance, ici, la lumière n'est pas visible, elle est située dans l'ultraviolet. C'est pour cette raison que dans les tubes fluorescents, il y a de la poussière (très polluante) déposée sur la face interne du tube. Vous ne comprenez pas ? Ce n'est pas grave, nous ne sommes pas ici pour expliquer le fonctionnement du tube fluo. (Allez, je vais vous le dire quand même ; l'énergie lumineuse non visible émise par l'argon excite à son tour les atomes de cette poussière qui, eux, vont émettre de la lumière visible. Suivant la nature de cette poussière, la lumière sera plutôt « froide », tirant sur le bleu ; ou « chaude » tirant sur le jaune rouge. La « température » d'une lumière se mesure en Kelvins ([Lien 01](#)), par analogie au rayonnement lumineux d'un corps noir chauffé à une certaine température.)

### 2.4. Observez le spectre



La lumière visible s'étend de l'infrarouge à l'ultraviolet, bornes non comprises. Bien entendu, ici, nous considérons que la lumière est une onde.

(1 nanomètre =  $10^{-9}$  mètre = 1/1 000 000 de millimètre.)

Si vous vous sentez plus à l'aise avec les fréquences, la lumière visible s'étend de  $4 \times 10^{14}$  à  $8 \times 10^{14}$  Hz (400 000 GHz à 800 000 GHz).

### 2.5. Et quelle vitesse ?

On a coutume de dire 300 000 km/s. C'est bien entendu faux, ça dépend du milieu dans lequel la lumière se propage. Ceci dit, les variations de vitesse restent

minimes ; elles peuvent tout de même apporter certaines perturbations suivant les conditions d'utilisation. Ce détail a son importance dans le cas de la fibre optique.

### 2.6. Dans quelle direction ?

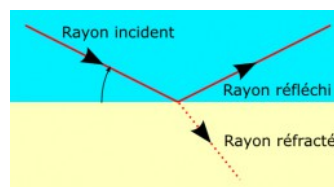
On a aussi coutume de dire qu'elle se propage en ligne droite. Vous l'avez deviné, ceci est également faux la plupart du temps. Ce n'est vrai qu'à la condition que le milieu dans lequel elle se propage soit homogène et isotrope, ce qui est rarement le cas. La preuve que c'est faux : les mirages existent.

Maintenant que toutes ces fausses bases sont données, passons à l'étape suivante.

### 2.7. Le changement de milieu

Que fait la lumière lorsqu'elle rencontre un obstacle ?

- Elle le traverse ?
- Elle est réfléchi par cet obstacle ?
- Elle est absorbée par cet obstacle ?
- Elle subit une combinaison de ces trois possibilités ?



Imaginons que le bleu soit de l'air et le jaune du verre. Un rayon de lumière qui vient de l'air vers le verre selon un angle d'incidence donné va :

- se réfléchir et retourner dans l'air, c'est le rayon réfléchi (sinon, les vitres de vos voisins ne vous éblouiraient jamais en réfléchissant le soleil) ;
- pénétrer dans le verre (sinon vous ne verriez rien à travers vos vitres ni à travers celles de vos voisins) en subissant une déviation de trajectoire (cette déviation ayant lieu deux fois, une vitre a généralement deux faces, elle n'est pas remarquable dans la plupart des cas, mais si l'on plonge un bâton dans de l'eau, ne semble-t-il pas se plier brutalement au passage dans l'eau ?) ;
- perdre un peu d'énergie dans l'aventure (sinon, les vitres ne chaufferaient pas au soleil).

Le bilan énergétique doit être nul, à savoir que l'énergie réfléchi plus l'énergie réfractée (transmise) plus l'énergie absorbée égale l'énergie incidente. (Loi de la conservation de l'énergie, sans laquelle le monde serait bien plus chaotique que ce qu'il est déjà.)

Le rapport entre l'énergie réfléchi et l'énergie transmise varie en fonction de l'angle d'incidence. Il existe un angle critique... Alors, ça dépend de la façon dont on le mesure. S'il est mesuré comme indiqué sur le schéma, lorsque cet angle devient inférieur à l'angle critique, il n'y a plus de rayon réfracté et, aux pertes par absorption près, la totalité du rayon incident est réfléchi. Ceci va être très important pour la suite.

L'angle de déviation entre le rayon incident et le rayon réfracté dépend de plusieurs choses :

- de la nature du dioptré (séparation entre les milieux) ; autrement dit, il dépend des deux

- milieux considérés ;
- le plus souvent, il dépend également de la longueur d'onde de la lumière incidente. Sinon, les arcs-en-ciel n'existeraient pas. Si les milieux ne sont pas dispersifs, alors il n'y a pas d'arc-en-ciel. Mais la plupart des milieux le sont (ceci aussi va d'ailleurs nous poser des problèmes).

### **2.8. C'est cohérent ou ça ne l'est pas ?**

Le plus souvent, ça ne l'est pas. Mais qu'est-ce que ça veut dire ? La lumière « normale », celle que l'on utilise habituellement (avec nos ampoules à incandescence, bientôt interdites), n'est pas cohérente. Les petits trains d'ondes lumineuses, émis par les électrons qui descendent les niveaux d'énergie, le font n'importe quand, de façon aléatoire. Il n'y a aucune cohérence dans la forme des ondes lumineuses, constituées d'une somme de petits trains d'ondes de même fréquence, mais émis avec une phase aléatoire. On ne peut donc pas observer une belle sinusoïde, comme on sait les faire en électricité par exemple. Ici, la théorie du photon arrange bien, c'est plus facile de parler d'un flux de particules que d'une onde sinusoïdale constituée de petits morceaux qui ne sont pas en phase.

De ce côté-là, le laser est bien intéressant parce qu'il fournit une lumière cohérente, ce qui lui donne des propriétés spécifiques que l'on ne va pas énumérer ici, mais dont on va relever quelques particularités :

- la lumière laser peut être considérée comme une onde à part entière, c'est une émission continue ;
- la lumière laser peut être concentrée sur un faisceau très fin et se propager non pas selon un cône, mais selon un cylindre ;
- la lumière laser est parfaitement monochromatique ;
- la lumière laser peut transporter beaucoup d'énergie.

Ceci nous suffira largement pour la suite.

## **3. La fibre optique**

### **3.1. Pour quoi faire ?**

Pour faire un tuyau dans lequel on peut faire passer de la lumière.

Nous avons vu que la lumière avait une certaine tendance à se propager en ligne droite. Pour transporter de l'information d'un point quelconque vers un autre point quelconque, ce n'est pas très pratique ; un tuyau, c'est mieux, ça peut prendre des virages.

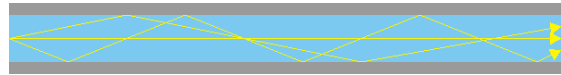
### **3.2. Comment faire ?**

Dans un premier temps, faisons simple : une lumière incohérente et pas forcément monochromatique, dans une fibre construite sans trop de précautions.

Le principe de base, c'est le coup du dioptre. La fibre de verre va être gainée d'un autre matériau tel que le dioptre ainsi formé soit avantageux pour nos besoins, à savoir :

- un angle critique le plus grand possible (tel que nous l'avons défini dans la page précédente), ceci afin de supprimer autant que possible tout rayon diffracté ;

- des absorptions d'énergie les plus minimales possible lors de la réflexion sur le dioptre.



Je vous entends me dire : « Oui, et dans les virages ? » Parce que c'est bien le but, prendre des virages. Dans les virages, c'est l'angle d'incidence qui va être malmené. Il faut s'arranger pour ne pas passer l'angle critique. Naturellement, tel que c'est dessiné ici, il y aura forcément des rayons qui arriveront dans le virage avec un angle trop grand et il y aura donc un rayon diffracté, perdu pour tout le monde.

Personne n'a jamais dit que les fibres optiques transmettaient la lumière sans pertes ! Et ce n'est d'ailleurs pas la seule source de pertes.

Il serait possible de faire des calculs pour définir, en fonction du rayon de courbure, du diamètre de la fibre et de l'indice de réfraction, les pertes d'énergie lumineuse... Nous n'allons pas le faire, mais en regardant le schéma ci-dessus, on comprend bien que, plus le diamètre du cœur de la fibre sera petit, plus on minimisera les risques d'un angle d'incidence trop grand. D'un autre côté, ce sera plus délicat de faire passer dans cette fibre une quantité de lumière donnée. C'est le même problème que dans un tuyau de plombier, à débit constant ( $m^3/s$ ), plus le diamètre sera petit, plus il faudra augmenter la pression.

### **3.3. Les problèmes qui arrivent ?**

#### **3.3.1. Le premier arrivé attend l'autre**

Ce n'est pas la peine de faire des calculs compliqués pour voir sur l'illustration que les divers rayons qui vont pénétrer dans la fibre vont suivre des chemins différents, plus ou moins longs suivant le nombre de réflexions subies. Comme ils vont tous à la même vitesse (du moins pour une longueur d'onde donnée), ils ne vont pas tous arriver à l'autre bout en même temps.

De plus, nous pouvons assister à des phénomènes d'interférences.

#### **3.3.2. Un kilomètre à pied ?**

La nature du verre (le verre, ou toute autre matière transparente fera l'affaire) et celle du dioptre font qu'il y a des pertes dans la fibre. Pertes dues à la turbidité du verre et pertes dues aux réflexions. Il ne faudra pas s'attendre à ce que la longueur utile d'une fibre optique soit infinie.

#### **3.3.3. La lumière se disperse**

Le verre et ses équivalents sont des milieux dispersifs. La vitesse de propagation va varier en fonction de la longueur d'onde. Si l'on introduit une lumière qui n'est pas monochromatique, on va récupérer en sortie plusieurs lumières « différentes » et ça ne va pas aider à reconstituer le signal.

#### **3.3.4. Et finalement ?**

Tous ces inconvénients vont imposer des limites d'utilisation :

- une « bande passante » maximale. Si l'on envoie des impulsions lumineuses, elles seront



récupérées avec une certaine distorsion et si cette distorsion devient trop grande, on ne pourra plus reconstituer l'information. Nous comprendrons mieux cet effet sur les illustrations qui suivent ;

- une longueur maximale. Il est assez compréhensible que, plus la fibre va être longue, plus ces perturbations vont être observées. Pour une performance attendue, il y aura une longueur maximale définie, en fonction des technologies utilisées.

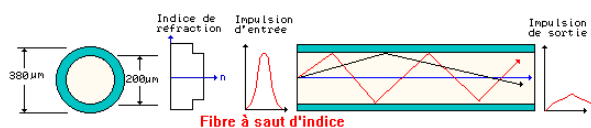
### 3.4. Les parades

Il va falloir construire des fibres capables de limiter le plus possible ces problèmes.

#### 3.4.1. La fibre multimode

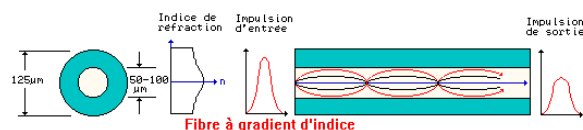
Dans cette famille, nous trouvons deux sous-catégories.

##### 3.4.1.1. La fibre à saut d'indice



C'est la plus « ordinaire ». Le cœur a un relatif gros diamètre, par rapport à la longueur d'onde de la lumière (de l'ordre du  $\mu\text{m}$  dans l'infrarouge). Tous les inconvénients vus plus haut se manifestent ici. Observez l'allure de l'impulsion de sortie, comparée à celle de l'impulsion d'entrée. Ce sont bien entendu des informations non quantitatives.

##### 3.4.1.2. La fibre à gradient d'indice

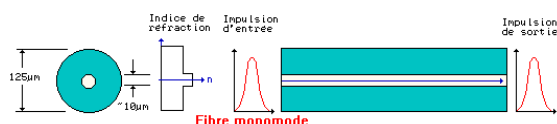


Ici, deux améliorations sont apportées :

- le diamètre du cœur est de deux à quatre fois plus petit ;
- le cœur est constitué de couches successives, à indice de réfraction de plus en plus grand. Ainsi, un rayon lumineux qui ne suit pas l'axe central de la fibre est ramené « en douceur » dans le droit chemin.

Comme vous pouvez l'observer, les résultats sont déjà de meilleure qualité.

#### 3.4.2. La fibre monomode



C'est le « top ». Le diamètre du cœur est très petit, les angles d'incidence le sont donc aussi. Les résultats sont excellents, mais, compte tenu de la faible section de cette fibre, seule la lumière laser est ici exploitable. Il n'y a pas de miracle, c'est la solution la meilleure, mais aussi la plus onéreuse.

Illustrations extraites du site : [Lien 02](#). Site que je vous invite par ailleurs à visiter si vous souhaitez en savoir plus sur ce domaine : [Lien 03](#). (Et sur d'autres domaines aussi.)

### 3.5. Quelques questions habituelles

#### 3.5.1. La fibre optique coûte-t-elle cher ?

Non. Par rapport au câble en cuivre, elle aurait même tendance à coûter moins cher, surtout avec l'envolée du prix des métaux. En revanche, la connectique et les convertisseurs d'énergie électrique/lumineuse et réciproquement à placer aux extrémités coûtent cher, très cher même, suivant les technologies mises en œuvre.

#### 3.5.2. La fibre optique est-elle bidirectionnelle ?

Oui. Cependant, on ne l'utilise souvent que dans un seul sens, pour simplifier les convertisseurs placés aux extrémités. Si l'on souhaite exploiter une fibre optique dans les deux sens, il faudra :

- utiliser des longueurs d'onde différentes pour chaque sens ;
- utiliser des extrémités capables de capter de la lumière pour la convertir en électricité ET émettre de la lumière en fonction d'un signal électrique. C'est réalisable, mais ça a un coût.

#### 3.5.3. Peut-on passer plusieurs informations différentes dans la même fibre et les récupérer intactes à l'autre bout ?

Oui, il y a même deux méthodes pour le faire :

- on utilise plusieurs longueurs d'onde lumineuse. Là aussi, il y a une incidence sur la complexité des équipements aux extrémités. C'est du multiplexage spatial, à rapprocher de la « large bande » sur le cuivre ou la HF ;
- on peut également faire du multiplexage temporel.

Ces techniques seront vues plus loin dans ce chapitre.

#### 3.5.4. Quels sont les principaux avantages de la fibre optique ?

- La fibre optique est totalement insensible aux rayonnements électromagnétiques dans lesquels nous baignons.
- L'atténuation du signal est inférieure à celle d'un conducteur électrique et les distances couvertes sans nécessité d'installer des amplificateurs sont bien plus grandes.
- La bande passante est généralement bien supérieure à celle que l'on peut obtenir avec un câble électrique.

#### 3.5.5. La fibre optique est-elle fragile ?

Pas particulièrement. C'est la connectique qui peut l'être. Le seul problème, c'est le rayon de courbure minimum qui la rend assez peu souple d'emploi pour les installations « volantes ».

#### 3.5.6. Quelles performances peut-on en attendre ?

D'une grosse centaine de mégabits par seconde,

comparable à ce que l'on sait faire avec du cuivre, au record actuel (à l'heure où ces lignes sont écrites) détenu par Alcatel : 10,2 Tbit/s (10 200 Gbit/s), sur une distance de 100 kilomètres. Un autre record : 3 Tbit/s (3000 Gbit/s), sur une distance record de 7300 kilomètres

### **3.5.7. Si la fibre a autant de qualités, pourquoi ne l'utilise-t-on pas plus ?**

Je vous le demande. Notez que l'on commence à en parler sérieusement, au moins dans les zones à forte concentration urbaine.

## **4. Mais comment font-ils ?**

Dans le principe, les méthodes sont simples. Ce qui l'est moins, c'est de maîtriser les technologies nécessaires. Mais pour comprendre le fondement, prenons un cas « simple ».

### **4.1. La paire téléphonique**

Tout le monde connaît ces deux bouts de fil de cuivre qui permettent de brancher un téléphone ? Voyons déjà ce que l'on peut faire avec.

#### **4.1.1. Bande passante du signal**

Dans le cas du transport de téléphonie analogique, nous allons véhiculer sur cette paire de cuivre un signal électrique analogique (dont la forme d'onde est analogue à celle du signal acoustique), dont la bande passante va de quelques Hertz à 4 kHz. Tout simplement parce que c'est largement suffisant pour garder un message vocal compréhensible. On a coutume de dire que l'homme est en mesure d'entendre des sons entre 20 Hz et 20 kHz. C'est une approximation, bien entendu. Cette bande passante couvre l'ensemble des fréquences audibles. Si l'homme pouvait, avec ses seules cordes vocales couvrir un aussi large spectre, ce serait amusant... Un chanteur lyrique est capable de couvrir à peine un peu moins de trois octaves (d'une fréquence à 8 fois cette fréquence).

#### **4.1.2. Bande passante du câble téléphonique**

Si par ailleurs, on essaye d'évaluer la bande passante de la paire téléphonique, on arrive généralement à faire passer des fréquences allant jusqu'au gigahertz sur une longueur « moyenne », entendons par là, la longueur moyenne qui relie un abonné à son centre de distribution. En pratique, c'est un peu plus complexe.

#### **4.1.3. Conséquence directe**

Il y a un formidable gaspillage de moyens. On sous-utilise abominablement les ressources d'une structure existante. Pourquoi ? Parce qu'utiliser 100 % des ressources de cette paire torsadée impose de mettre en œuvre des technologies complexes, donc chères, et qu'il faut les rentabiliser.

#### **4.1.4. xDSL**

Aujourd'hui, tout le monde court après les moyens de transport d'information numérique, connexion à l'Internet oblige. Les technologies deviennent rentables et on les exploite. Ici, la solution consiste à utiliser des fréquences situées au-dessus de 4 kHz, pour qu'elles n'interfèrent pas avec la téléphonie, et de les moduler pour leur faire transporter de l'information. Voyez le chapitre sur la bande

passante pour plus de détails sur ces techniques. La moralité est que l'on peut obtenir une connexion Internet à « haut débit » (ADSL) qui va passer par la ligne téléphonique, sans perturber la téléphonie, le tout pour un prix acceptable, simplement en optimisant les ressources d'une installation existante.

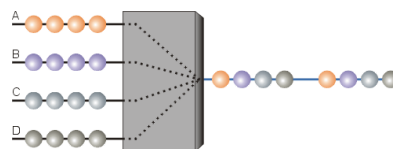
## **4.2. La fibre optique**

Sur la fibre optique, on va essayer aussi de trouver des technologies, qui, pour une qualité de fibre donnée, vont chercher à optimiser le flux d'informations dans cette fibre. En effet, la fibre optique, même la plus rudimentaire (saut d'indice) dispose d'une bande passante au moins égale à celle que l'on peut espérer d'une paire torsadée. Les fibres de type « gradient d'indice » sont déjà beaucoup plus performantes. Quant aux fibres « monomodes », elles disposent d'une bande passante incomparable.

Il existe « grosso modo » deux méthodes.

### **4.2.1. Multiplexage temporel**

Là encore, prenons un cas simple.



D'un côté, nous avons quatre lignes à faible débit A, B, C et D, disons, 640 kbit/s. De l'autre côté, nous avons une fibre optique qui pourrait passer facilement 100 fois plus... Autrement dit, alors que la ligne A, par exemple, va mettre une seconde à déverser 640 kbit, la fibre optique va faire passer ces 640 kbit en 1/100 de seconde, et va attendre 99/100 de seconde le paquet suivant en provenance de la ligne A.

Ici l'on va tout simplement utiliser un multiplexeur temporel, qui va accumuler des paquets de données provenant des lignes A, B, C et D et les passer séquentiellement sur la fibre optique.

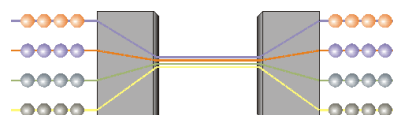
Dit autrement, vous avez quatre petites routes, où les voitures roulent pare-chocs contre pare-chocs. Ces quatre routes débouchent sur une autoroute à 10 voies. Les routes sont saturées, mais l'autoroute peut encore accepter beaucoup d'autres voitures.

Ce type de multiplexage s'appelle TDM (Time Division Multiplexing).

Dans cette approche, nous utilisons une fibre optique avec une seule source lumineuse. C'est peut-être dommage, parce que cette fibre, à l'image d'un câble de cuivre, peut faire passer plusieurs fréquences (longueurs d'onde), donc plusieurs couleurs.

### **4.2.2. Multiplexage spatial**

Si tôt dit, si tôt fait. Nous allons utiliser plusieurs lasers de couleurs différentes. Ces faisceaux lasers pourront voyager dans la fibre et être récupérés individuellement à l'autre bout, grâce à de « simples » filtres optiques.



Une fibre optique peut facilement transporter des longueurs d'onde comprises entre 1530 nm et 1565 nm, nous sommes dans l'infrarouge (l'illustration fait apparaître des couleurs pour la compréhension) et sur de la fibre monomode. 35 nm d'écart, ça ne paraît pas beaucoup. Oui, mais comme on sait séparer deux ondes lumineuses si la différence de longueur d'onde est de 0,8 nm et même 0,4 nm, alors, on peut passer dans la même fibre de 43 à 87 « lumières » différentes. Cette méthode s'appelle : DWDM (Dense Wavelength Division Multiplexing). Si l'on considère que l'on peut passer sans problèmes 2,5 Gbit/s sur chaque canal...

Avec cette méthode, il est même possible d'utiliser certains canaux dans un sens et d'autres canaux dans l'autre, ce qui permet de faire du « full duplex » avec une seule fibre.

#### 4.2.3. Encore plus fort

Paris, le 21 mars 2001 - Alcatel (Paris : CGEP.PA, NYSE : ALA), leader mondial des réseaux optiques intelligents, a établi deux nouveaux records du monde pour des transmissions DWDM multitérabits. Le Groupe a d'une part franchi la barre mythique des 10 Tbit/s (10 000 Gbit/s), établissant le record mondial absolu de capacité de transmission sur une fibre optique. Alcatel a d'autre part réalisé une transmission record de 3 Tbit/s (3000 Gbit/s) sur une distance transocéanique de 7300 kilomètres (Source Alcatel).

#### 4.2.4. Mais ?

Cette technique emploie :

- de la fibre mono mode ;
- une (des) source(s) lumineuse(s) laser, ce qui est obligatoire avec ce type de fibre.

La fibre mono mode a un diamètre de l'ordre de 10 µm (1/100 de millimètre). Ce n'est pas bien gros et l'on devine que les technologies nécessaires pour « enfile » un rayon laser dans cette fibre ne sont pas simples. A fortiori s'il faut en faire passer plusieurs. Et il faut aussi récupérer les signaux à l'autre bout.

En d'autres termes, les performances de cette technologie n'ont d'égal que son prix. Tout le problème consiste donc à choisir la technologie la mieux appropriée à ses besoins présents et futurs, pour ne pas se retrouver avec un réseau :

- ruineux parce que la technologie utilisée, trop chère, ne peut être rentabilisée ;
- ruineux parce que la technologie utilisée, trop peu performante pour une raison de prix, ne pourra pas assurer l'inévitable croissance ultérieure, autrement qu'en multipliant les équipements de bout en bout du réseau.

#### 4.3. Les sources lumineuses

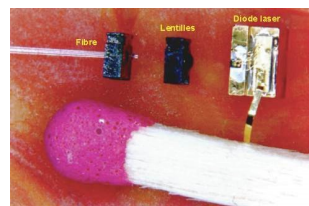
Le plus souvent, ce sont des diodes électroluminescentes. Il en existe de toutes sortes. Il en existe même qui sont capables d'émettre un faisceau laser.

Ces dispositifs ne sont pas énormes, témoin l'illustration ci-dessous. L'allumette donne l'échelle.

#### 4.4. Les capteurs de lumière



Les capteurs ne sont pas plus gros, ce sont des photodiodes, diodes sensibles à la lumière, qui permettent assez simplement de convertir une variation d'intensité lumineuse en variation de courant électrique.



#### 5. Conclusion

La fibre optique représente assurément le meilleur moyen actuel pour transporter de très hauts débits d'informations numériques et les besoins dans ce domaine vont probablement augmenter très fortement dans un avenir proche. Il est vraisemblable que la demande concernant un simple accès Internet, d'ici quelques années, sera identique à ce que l'on attend aujourd'hui d'un réseau local (100 Mbit/s au moins). Dans ces conditions, le panorama de l'information aura complètement changé. La téléphonie, la radio, la télévision et les transferts de données « informatiques » seront assurés par la même connexion, les interpénétrations de ces divers moyens d'informations seront beaucoup plus grandes, c'est du moins un scénario tout à fait réaliste.

Les locations de films vidéo, les chaînes de télévision dédiées à la cinématographie et la vente de CD audio seront sans doute les premières activités remises en question.

Avec des débits équivalents à ceux d'un réseau local, l'Internet va évoluer considérablement. Au choix, suivant les groupes de pression et les intérêts économiques mis en jeu, vers :

- un Internet de plus en plus « peer to peer » où le contenu va refluer vers la périphérie du réseau, ce qui serait dans le droit fil de la philosophie initiale de l'Internet, mais irait à l'encontre de bien des intérêts économiques en place ;
- un Internet de plus en plus « minitelisé », où petit à petit, le contrôle de l'information qui circule sera remis entre les mains d'entités spécialisées vers le cœur du réseau, la périphérie du réseau ne devenant plus que consommatrice, comme nous l'avons connu avec le Minitel, exception nationale dont la France fut très fière, mais qui n'a jamais réussi à convaincre quiconque en dehors de nos frontières.

**Note :** la conclusion peut certes paraître hors sujet (encore que, pour quelles raisons nos opérateurs marchent-ils à vitesse aussi forcée vers des débits toujours plus grands ?), mais elle mérite tout de même que l'on y réfléchisse.

Retrouvez l'article de Christian Caleca en ligne : [Lien 04](#)

### Automatisez sous Mac OS X 10.8 Le supplément pour Mountain Lion

Automatiser, oui... mais comment ? Grâce à Automator, le bien nommé, outil intégré à votre Mac lors de son achat, qui vous permettra de piloter plus efficacement votre ordinateur.

À l'aide de 28 tutoriels vidéo, Sylvain Gamel présente et explique les nouveautés et changements introduits par OS X 10.8, Mountain Lion.

Ce livre est une mise à jour de l'édition précédente qui avait été rédigée en se basant sur OS X 10.7 Lion.

Ce livre ne reprend pas les bases d'Automator telles qu'elles avaient été décrites dans « Automatisez pour Mac ».

Il les complète, le plus souvent en utilisant des vidéos.

L'écrit a bien entendu une large place, mais il vient en support des 28 tutoriels vidéo qui détaillent les exemples présentés.

Ainsi, même si vous avez déjà lu le premier ouvrage, celui-ci vous aidera probablement à aller plus loin en abordant des sujets différents et avec un support visuel original.

**Chapitre 1** : Du Lion au Lion des Montagnes. Ce premier chapitre vous présente brièvement les changements introduits dans OS X 10.8.

**Chapitre 2** : L'application Automator. Dans le second chapitre, Automator est un peu détaillé, son interface rapidement décrite. Y est également présenté le fonctionnement du mode pas à pas, ainsi que les différents types de processus.

**Chapitre 3** : Les principales actions. Dans ce chapitre, vous verrez quelques actions qui ont été impactées par la transition 10.7 vers 10.8. Vous y trouverez également quelques exemples complémentaires, comme l'utilisation de bases de données SQL avec SQLite.

**Chapitre 4** : Utilisation avancée. Ce dernier chapitre va vous permettre d'aller un peu plus en profondeur. Des clés pour vous aider à résoudre les principaux problèmes de mise au point d'un processus vous sont données, ainsi que l'utilisation détaillée des variables et leur personnalisation. Enfin, dans une dernière partie, vous verrez comment écrire vos propres actions pour Automator.

#### Critique du livre par Aurélien Gaymay :

Ce livre est très bien adapté pour les utilisateurs débutants et expérimentés avec ou sans connaissance en programmation. Vous allez découvrir ou redécouvrir en détail l'utilisation de l'application Automator qui est fournie par défaut sur tous les Mac et qui est méconnue du grand public.

Grâce à cette application et à ce livre, vous allez pouvoir créer de petites applications en quelques clics vous permettant d'automatiser des tâches répétitives et de gagner de temps. Pour les utilisateurs avancés, ils vont pouvoir coupler cette application à d'autres langages comme à l'AppleScript, Perl, Python, Ruby ou d'interagir avec une base de données.

L'arrivée des vidéos dans ce livre électronique est un petit plus qui permet d'avoir un aperçu des automates qui sont présentés, car les images c'est bien, mais en vidéo c'est encore mieux. Une introduction très sympathique avec une vidéo montrant les nouveautés qui différencient 10.8 à 10.7 et les changements de noms d'applications comme l'application Agenda.

Avec une connaissance du langage AppleScript, vous dompterez votre Lion des Montagnes aux doigts et à l'oeil. De plus, ce livre est maintenant disponible en version iBooks? Alors, tous à vos iPad...

*Retrouvez l'ensemble des critiques de livres de la rubrique Mac en ligne : [Lien 05](#)*

## Les derniers tutoriels et articles

### Les actions de dossier avec AppleScript

L'utilisation des actions de dossiers sur Mac OS peut être extrêmement productive, à condition de bien les maîtriser.

#### 1. Préambule

En se promenant sur la toile, vous verrez beaucoup de questions à ce propos, certains se plaignant même de dysfonctionnements. Autant le dire tout de suite, les actions de dossiers fonctionnent... à condition de connaître leurs règles et surtout leurs limites ! Il est possible d'en contourner certaines, mais pas toutes : autant le savoir avant de se lancer.

Bien qu'il soit aussi possible (et très facile !) d'utiliser Automator pour les actions de dossiers, cet article porte uniquement sur leur programmation en AppleScript.

Tous les exemples de cet article ont été testés sur un iMac27 (i7, Snow Leopard). De nombreux tests (mais pas tous !) ont été faits sur un iMac20 (Core duo, Snow Leopard) sur un Power Mac (G5 bipro, Tiger) et sur un iMac24 (i5, Mountain Lion). Je n'ai constaté aucune



différence d'exécution (hormis la vitesse !), ce qui permet de penser que tout ce qui suit fonctionne sur tous les systèmes, au moins depuis Tiger.

Tous les codes sont libres d'utilisation.

Les actions de dossier permettent de déclencher automatiquement des scripts, en général après l'ajout de fichiers dans un dossier, mais aussi liés à d'autres événements sur le dossier.

Par exemple, glisser les fichiers images de formats différents dans un dossier peut les convertir automatiquement en format JPG de taille identique. Ou encore de changer leur nom avec un incrément ou avec la date de leur ajout dans le dossier.

Il est possible de limiter les types, faire des conversions vidéo, texte ou PDF, les envoyer vers un serveur distant, les imprimer... Bref tout ce que vous pouvez imaginer. Il existe des actions toutes faites, vous pouvez en créer avec Automator, mais aussi écrire vos propres scripts. Dans ce dernier cas, cet article est fait pour vous aider.

## 2. Création des actions de dossiers

### 2.1. Création et enregistrement

Avec l'AppleScript Editeur (dossier Utilitaires), il faut créer son script en l'entourant du « handler » adéquat qui dépend de l'action.

Exemple de script pour ajouts de fichiers :

```
on adding folder items to Mon_Dossier after
receiving Liste_Fichiers
```

```
    -- Insérer votre script ci-dessous
    -- La variable Liste_Fichiers contient la
liste des noms de fichiers ajoutés
    -- Mon_Dossier contient le nom du dossier
dans lequel les fichiers sont ajoutés.
```

```
end adding folder items to
```

Enregistrez votre programme sous forme de script (extension « scpt ») dans le chemin : Bibliothèque/Scripts/Folder Action Scripts.

Si la bibliothèque est celle de l'utilisateur, l'action n'est disponible que pour cet utilisateur. Pour la rendre disponible à d'autres utilisateurs de la machine, il faut créer le même chemin à partir de la bibliothèque située à la racine du disque (avec les droits administrateurs).

Attention, ce chemin n'existe pas par défaut. Il faut éventuellement créer les dossiers « Scripts » et « Folder Action Scripts » soi-même. Le dossier bibliothèque n'étant plus visible directement dans Mountain Lion, il faut utiliser le menu du « Finder Aller... » en appuyant sur la touche Alt pour y avoir accès.

Pour faciliter l'installation d'un script, en particulier si vous le distribuez à des utilisateurs débutants, le script ci-dessous enregistre automatiquement votre action à la bonne place en prenant soin de créer les dossiers nécessaires. Il pourrait être complété par la création du lien avec un dossier (voir ajout d'action par un script - chapitre III.B).

#### Ajout d'un script dans la bibliothèque utilisateur

```
-- Installation d'un script pour Action de
dossier dans la bibliothèque
```

```
-- © PBell 2011 09
```

```
set F_Script to "Scripts"
set F_Action to "Folder Action Scripts"

-- Sélection du script à installer
tell application "Finder"
    set Mon_Script to (choose file with prompt
"Sélectionner le script à installer" without
invisible) as alias
    if name extension of Mon_Script is not "scpt"
then
        display dialog "Désolé, le fichier " &
(name of Mon_Script) & -
            " n'est pas un script." & return &
return & "L'installation a échoué" buttons
{"Fin"}
        return
    end if

    set Ma_Bibli to path to library folder from
user domain as alias
    -- Le dossier "Scripts" dans la Bibliothèque
utilisateur existe ? Sinon, on le crée
    if not (exists (folder F_Script of Ma_Bibli))
then
        make new folder at Ma_Bibli with
properties {name:F_Script}
    end if

    -- Le dossier "Folder Action Scripts"
existe ? Sinon, on le crée dans le dossier
Scripts
    set Doss_Script to ((Ma_Bibli as string) &
F_Script & ":") as alias
    if not (exists folder F_Action of
Doss_Script) then
        make new folder at (folder F_Script of
Ma_Bibli) with properties {name:F_Action}
    end if
    delay 1 -- parfois le Finder prend un peu de
temps !

    -- Copie du script à l'emplacement voulu
    try
        copy file Mon_Script to (folder F_Action
of Doss_Script)
    on error
        display dialog "Erreur lors de la copie
du fichier" buttons {"Abandonner"}
    end try
end tell
```

Les plus perfectionnistes peuvent modifier et enregistrer ce script comme une application et insérer, dans le paquet qui en résulte, le fichier du script à ajouter. Ce faisant, un simple double-clic sur l'icône de l'application installera tout sans rien demander à l'utilisateur !

### 2.2. Les types d'action de dossier

Il y a cinq types d'actions de dossiers :

- « on Opening folder *Mon\_Dossier* » : appelée après ouverture de la fenêtre du dossier ;
- « on Closing folder window for *Mon\_Dossier* » : lorsque la fenêtre du dossier se ferme ;
- « on moving folder window for *Mon\_Dossier* from *Ancienne\_Position* » : quand la fenêtre est déplacée ou redimensionnée ;
- « on removing folder items from *Mon\_Dossier* after losing *Liste\_Fichiers* » : lorsque des fichiers

sont supprimés du dossier ;

- « on adding folder items to *Mon\_Dossier* after receiving *Liste\_Fichiers* » : lorsque des fichiers sont ajoutés au dossier.

*Mon\_Dossier* est l'alias du dossier concerné.

*Ancienne\_Position* est de type « bounding rectangle » (les coordonnées du rectangle de la fenêtre).

*Liste\_Fichiers* est de type liste et contient un ou plusieurs fichiers concernés avec leur chemin d'accès (attention, voir chapitre V concernant cette variable).

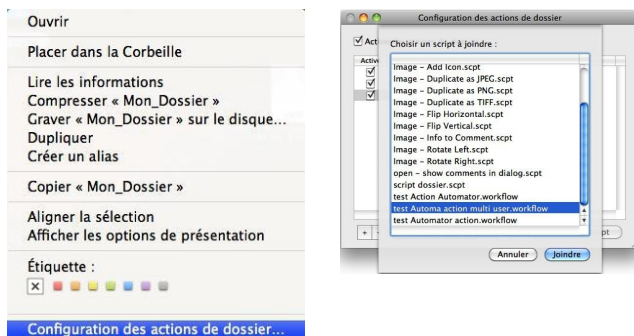
La dernière action, « adding », est, de loin, la plus souvent utilisée. C'est aussi celle qui peut poser le plus de problèmes. C'est la raison pour laquelle nous allons nous y intéresser en détail.

On constate qu'il n'y a malheureusement pas d'action déclenchée lors de la modification d'un fichier. Il existe d'autres méthodes pour cela, mais pas d'action de dossier !

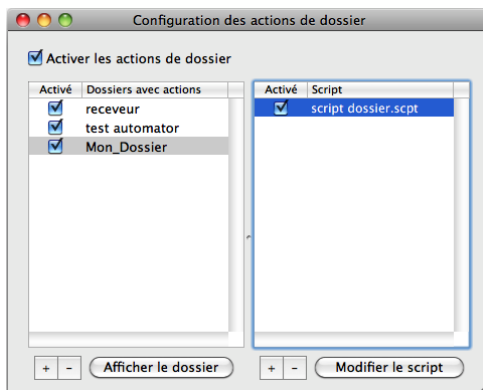
### 3. Comment assigner les actions à des dossiers

#### 3.1. Assignment manuelle

Une fois le script créé et enregistré en bonne place, le plus simple est de sélectionner le dossier dans le Finder, faire un clic droit sur celui-ci, et choisir « Configuration des actions de dossier » (menu ci-dessous à gauche). Sur la liste qui apparaît (ci-dessous à droite), sélectionnez votre action de dossier, puis validez avec le bouton « Joindre ».



La liste des scripts de dossier intègre toutes les actions Automator et AppleScript provenant aussi bien de la bibliothèque utilisateur que de la bibliothèque racine du disque.



Voici l'écran d'assignation tel qu'il devrait être.

Ne pas oublier d'activer les actions de dossier dans la case en haut à gauche de cette fenêtre.

Cette validation n'est nécessaire que la première fois où

vous utilisez les actions de dossier.

Il est possible d'assigner plusieurs actions à un même dossier : sélectionnez votre dossier dans la colonne de gauche, puis utilisez les boutons « + » et « - » en bas de la colonne de droite pour ajouter ou supprimer des actions de ce dossier. Compte tenu du mécanisme interne de déclenchement des actions, je déconseille aux débutants l'utilisation de plus d'une action par dossier.

Il est aussi possible d'assigner des actions à des dossiers via des scripts (voir assignation en masse).

Pour les curieux, les liens entre dossiers et actions sont dans le fichier utilisateur/bibliothèque/Preferences/com.apple.FolderActions.plist, éditable avec l'utilitaire Property List Editor fourni avec le kit de développement Xcode d'Apple ou bien un éditeur de texte quelconque en prenant soin de conserver les balises.

Normalement, vous n'avez pas à vous occuper de ce fichier. Cependant, ce fichier peut devenir assez gros et entraîner des lenteurs, car il n'est pas mis à jour automatiquement lors de la suppression d'un dossier.

Pour le maintenir propre, il faut s'astreindre, avant de supprimer un dossier lié à une action, à supprimer ce lien avec les boutons « - » côté script d'abord (colonne de droite), puis l'entrée sur le dossier lui-même (colonne de gauche), après avoir sélectionné les éléments dans leur liste respective (voir fenêtre ci-dessus).

#### 3.2. Assignment en masse et propagation aux sous-dossiers

L'action de dossier, Adding, n'est déclenchée que si un fichier est ajouté dans ce dossier. Elle n'est pas activée lors de l'ajout d'un fichier dans un sous-dossier de ce dossier. Il est cependant possible de contourner cette limitation avec des scripts.

Imaginons que vous ayez un dossier parent contenant de nombreux sous-dossiers :

- un script peut ajouter une action de dossier à tous les sous-dossiers d'un dossier parent sélectionné ;
- une action de dossier sur le dossier parent peut assigner automatiquement un script d'action à tout nouveau sous-dossier ajouté.

Les scripts ci-dessous correspondent aux deux méthodes, la seconde étant plus souple. Ils sont suffisamment commentés pour que vous puissiez les adapter à vos besoins.

Ce script gère les erreurs en cas d'ajout (si l'action existe déjà pour certains des sous-dossiers) et en cas de suppression (si l'action n'existe pas dans tous les sous-dossiers). Il contient les instructions sur les actions (création, lecture et suppression) qui montrent les syntaxes à utiliser dans chaque cas.

```
-- Ce script : ajoute ou supprime la même action de dossier à tous les sous-dossiers d'un dossier parent
-- © PBell 2010-05 & 2013-01
-- Demande à l'utilisateur s'il veut ajouter ou supprimer les actions des sous-dossiers du dossier parent
```

```
set Reponse to display dialog "Voulez-vous ajouter ou supprimer des actions de dossier sur tous les sous-dossiers d'un dossier parent ?" &
```

```

return -
    & return & "(limité à un seul sous-niveau de
dossier)" buttons {"Annuler", "Ajouter",
"Supprimer"}
set Choix to button returned of Reponse

-- Récupération du chemin par défaut des Folder
Action Scripts
tell application "System Events"
    set Script_Folder to (path to Folder Action
scripts folder from user domain) as string
end tell

tell application "Finder"
    -- Sélection du dossier parent
    set Dos_Parent to (choose folder with prompt
"Sélectionner le dossier parent")
    -- Récupération des sous-dossiers du dossier
parent
    set Sous_Dossiers to every folder of folder
Dos_Parent

    -- Sélection du script à ajouter ou supprimer
(dans le dossier folder action scripts!)
    set Mon_Script to (choose file with prompt -
    "Sélectionner le script à
assigner/supprimer" default location
Script_Folder as alias)

    if Choix = "Ajouter" then
        set Mon_Texte to "Êtes-vous certain de
vouloir assigner le script à tous ces dossiers ?"
    else
        set Mon_Texte to "Êtes-vous certain de
vouloir supprimer le script de tous ces
dossiers ?"
    end if
    display dialog "Le dossier parent '" & (name
of Dos_Parent) & "' contient " & -
    (count of Sous_Dossiers) & " sous-
dossiers." & return & return & Mon_Texte

    -- boucle sur chaque sous-dossier du dossier
parent
    repeat with Un_Dossier in Sous_Dossiers
        tell application "System Events"
            if Choix = "Ajouter" then
                -- Ajout de l'action avec Try au
cas où cette action existe déjà
                try
                    attach action to Un_Dossier
as alias using (Mon_Script as string)
                end try
            else
                -- Suppression de l'action de
dossier : on lit toutes les actions
                set Mes_Actions to attached
scripts (Un_Dossier as alias)
                -- On boucle juste pour montrer
la syntaxe de lecture des actions
                repeat with Mon_Action in
Mes_Actions
                    if name of Mon_Action = name
of Mon_Script then
                        try
                            remove action from
(Un_Dossier as alias) -
                                using action name
                                (name of Mon_Action)
                        end try
                    end if
                end repeat -- fin boucle sur les
actions

```

```

end if
end tell
end repeat -- fin de boucle sur les sous
dossiers

display dialog "Action effectuée sur les
sous-dossiers" buttons {"OK"} -
default button 1 giving up after 5
end tell

```

Ce script gère un seul niveau de sous-dossier. Pour gérer plusieurs niveaux (sous-sous-dossiers...), il doit être adapté et faire appel à la récursivité, ce qui sort du cadre de cet article.

Ce second script est une action de dossier à assigner au dossier parent, avant création de sous-dossiers. Ensuite, dès l'ajout d'un sous-dossier, ce dernier sera automatiquement associé à une action de dossier prédéfinie (qui doit donc exister en bonne place).

```

-- Ce script est déclenché par une action de
dossier "add item" sur le dossier parent
-- si l'action est un l'ajout d'un sous-dossier,
-- alors ce script assigne une action de dossier
prédéfinie à ce nouveau sous-dossier
-- © PBell 2012-05

on adding folder items to Mon_Dossier after
receiving Liste_Fichiers

    -- Mettre ici le nom du script à ajouter à la
création d'un sous-dossier
    -- Ce script doit exister dans votre Folder
Action Script !!
    set Script_sous_Dossier to "Mon script de
sous dossier.scpt"

    -- Récupération du chemin par défaut des
Folder Action Scripts dans le domaine utilisateur
    tell application "System Events" to set
Dossier_Scripts to -
        (path to Folder Action scripts folder
from user domain) as string
        set Mon_Script to (Dossier_Scripts &
Script_sous_Dossier)

    -- Vérification que le script prédéfini
existe. Sinon, on quitte le script !
    tell application "Finder"
        if not (exists file Mon_Script) then
            return

            -- Boucle sur chaque item ajouté = un ou
plusieurs sous-dossiers)
            repeat with Un_Item in Liste_Fichiers
                if (kind of Un_Item) is "dossier"
                then
                    -- C'est un dossier, on ajoute
l'action de dossier
                    tell application "System Events"
to attach action to Un_Item as alias -
                        using (Mon_Script as string)
                    end if
                end repeat -- fin de boucle sur les items
ajoutés
            end tell
        end adding folder items to

```

#### 4. Règles de programmation des actions de dossier

Contrairement aux scripts habituels et aux processus Automator, les actions de dossier ne sont pas appelées par l'utilisateur, mais par le système lui-même.

Le mécanisme exact est hors de notre propos, mais ce qu'il faut retenir pour simplifier c'est que l'action de dossier se déroule en arrière-plan, et ce, même si vous voyez certains de ses effets. En conséquence, toute erreur rencontrée lors de l'exécution sera invisible, le script s'arrêtera sans aucun message visible (mis à part dans l'utilitaire Console). Vous ne saurez pas ce qui a eu lieu, ce qui n'a pas eu lieu, ni quand il s'est arrêté ! Il est donc particulièrement important d'écrire des scripts qui gèrent correctement toutes les erreurs possibles, faute de quoi, vous aurez des surprises (mauvaises, bien sûr !).

Les blocs « try/end try » et autres « if » vous aideront selon ce que vous voulez faire. Pour les scripts un peu complexes, je vous conseille fortement de les développer en mode script standard pour tracer toutes les erreurs possibles avec l'Éditeur AppleScript.

Pensez juste à ajouter, au début, une sélection d'un dossier et de fichiers de ce dossier pour simuler l'action de dossier comme indiqué ci-dessous :

```
-- Simulation de l'action de dossier d'ajout de
fichiers sur un script standard

-- Sélection du dossier
set Mon_Dossier to ((choose folder with prompt
"Choisissez le dossier test :" without
invisibles) ~
    as alias) as string

-- Sélection des fichiers ajoutés
set Liste_Fichiers to choose file with prompt
"selectionnez fichiers ajoutés" default location
~
    ((Mon_Dossier) as alias) with multiple
selections allowed

-- Insérez votre script ici
-- Vous pouvez exécuter ce script dans l'Éditeur
AppleScript pour le mettre au point
```

Une fois le script mis au point, vous supprimez les instructions de sélection du dossier et des fichiers et vous encadrez votre script par les « handler » adéquats comme ci-dessous :

```
on adding folder items to Mon_Dossier after
receiving Liste_Fichiers

    -- votre script ici, tel qu'il a été mis au
point (sans la sélection du dossier et des
fichiers bien sûr !)

end adding folder items to
```

Enfin, une petite précision : si votre script doit se déclencher à l'ajout d'un élément dans le dossier, et qu'il prévoit l'ajout d'un fichier de trace ou d'un sous-dossier dans ce même dossier, pensez à traiter la récursivité que cela va engendrer. En effet, votre script, en ajoutant un élément, va lui-même déclencher une action de dossier... donc il va s'appeler !

Votre script doit donc prévoir de ne rien faire lorsqu'un élément qu'il génère est ajouté au dossier. À titre

d'exemple, le script ci-dessous prévoit d'ajouter un sous-dossier (s'il n'existe pas déjà) dans lequel sont enregistrés les fichiers dont l'extension n'est pas retenue.

```
(* Script d'action de dossier :

Lors de l'ajout de fichiers dans le dossier, on
vérifie leur type
Si le type est incorrect, le fichier sera copié
dans un sous-dossier de rejet

Si le dossier de rejet n'existe pas déjà, on le
crée dans le dossier.
*)

on adding folder items to Mon_Dossier after
receiving Liste_Fichiers

    --Nom du sous dossier qui recevra les
fichiers dont l'extension est incorrecte
    set Dos_Rejet to "Mes_rejets"
    -- Liste des extensions de fichier acceptées
    set liste_Ext to {"jpg", "txt"}

    tell application "Finder"
        -- Pour éviter que l'ajout du dossier
Dos_Rejet ne crée aussi un événement,
        -- On teste ce qui est ajouté et,
éventuellement, on ne fait rien !
        if name of first item of Liste_Fichiers
is Dos_Rejet then return

        -- Vérifie si le sous dossier existe
déjà, sinon, le crée
        if not (exists folder Dos_Rejet of
Mon_Dossier) then
            make new folder at Mon_Dossier with
properties {name:Dos_Rejet}
            -- Attention, cet ajout génère un
event add folder items !! -> d'où le premier test
!
        end if

        -- On crée une liste de fichiers à
rejeter vers le sous-dossier de rejet
        set Liste_Rejet to {}
        repeat with Mon_Item in Liste_Fichiers
            if name extension of Mon_Item is not
in liste_Ext then
                set end of Liste_Rejet to
Mon_Item
            end if
        end repeat

        -- On transfère les fichiers incorrects
dans le dossier de rejet
        move Liste_Rejet to folder Dos_Rejet of
Mon_Dossier as alias

        display dialog "Opération terminée"
    end tell
end adding folder items to
```

#### 5. Fonctionnement spécifique de l'action « on adding folder items » et ses limites

L'instruction est : *on adding folder items to Mon\_Dossier after receiving Liste\_Fichiers.*

Le script doit se terminer par : *end adding folder items to.*



## 5.1. Que contient réellement le paramètre Liste\_Fichiers ?

Compte tenu des termes utilisés, notamment le mot « after », on peut s'attendre à ce que l'ajout de n fichiers simultanément dans un dossier déclenche le script avec Liste\_Fichiers contenant les n fichiers ajoutés, donc après l'ajout complet.

C'est parfois le cas, mais pas toujours ! Tout dépend du temps... mis par le système pour faire cet ajout.

En pratique tout ce passe avec deux processus parallèles, non synchronisés :

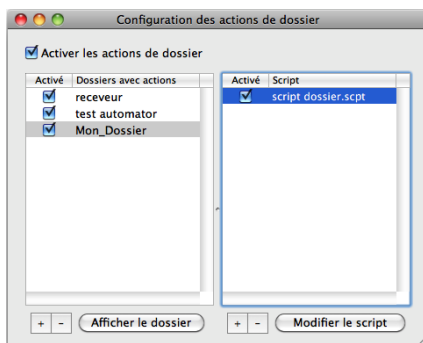
1. copie ou transfert des fichiers sélectionnés dans le dossier ;
2. temporisation interne, puis lancement du script d'action de dossier.

Lorsque le processus 1 est fini avant le 2, Liste\_Fichiers recevra TOUS les fichiers ajoutés.

Si le processus 1 n'est que partiellement fini, le processus 2 ne recevra qu'une partie des fichiers ajoutés...

Et les autres ? Eh bien ils feront l'objet d'un nouvel appel au script d'action de dossier ! Plus tard, après... au bout d'un *certain temps* pour paraphraser Fernand Raynaud !

Exemple avec ajout de quatre fichiers qui vont déclencher trois actions :



Chaque action déclenchée aura des paramètres différents : la première avec le fichier 1, la seconde contiendra les fichiers 2 et 3, car au début de cette action, le fichier 2 est ajouté et le 3 a commencé sa copie. Enfin la dernière action avec seulement le fichier 4.

On s'attendait pourtant à seulement une action avec les quatre fichiers !

Le problème se complique pour trois raisons :

- la vitesse du processus 1 dépend des supports physiques, de la taille des fichiers et du type d'ajout (copie ou transfert) ;
- la vitesse du processus 2 et sa temporisation dépendent de la machine et de son OS. Les variations sont cependant faibles ;
- le processus 2 ne se déclenche pas « after » la copie, mais peu de temps après le début de cette copie (aïe, aïe ! voir paragraphe V.B).

Il est facile de comprendre que la vitesse de l'ajout dépend des caractéristiques des disques/volumes d'origine et de destination (celui du dossier). Il faut se souvenir qu'elle va aussi dépendre de la méthode : copie ou transfert, via le Finder, une application, une commande shell Unix ou une autre machine en réseau.

Rappel :

La copie garde le fichier à la source et l'ajoute à la

destination. Elle s'exécute soit avec la commande cp en Unix, soit en glissant les fichiers d'un volume à l'autre sur le Finder, soit en glissant les fichiers entre deux emplacements d'un même volume avec la touche Alt.

Le transfert ne peut se faire qu'entre deux dossiers d'un même volume, soit avec la commande mv en Unix, soit avec un simple glisser sur le Finder. En fait, cette commande ne fait qu'écrire un nouveau chemin pour le fichier, sans réécrire ce dernier.

Lors d'un transfert, il y a de fortes chances que la vitesse soit telle que l'action de dossier reçoive la liste de tous les fichiers, car le transfert est très rapide ! Lors d'une copie, la réponse dépendra de la taille des fichiers et des vitesses des matériels.

Pour s'en convaincre, rien de tel qu'un exemple pratique : prenez un dossier D1 avec quatre fichiers : deux petits de 4Ko, deux gros de 2Go chacun au moins (des vidéos par exemple).

Prenez un dossier D2, sur le même volume/disque auquel vous assignez l'action de dossier contenant le script ci-dessous qui se contente d'afficher, pendant une seconde après le lancement de l'action, le nombre de fichiers reçus par l'action et le nom du premier :

```
on adding folder items to Mon_Dossier after
receiving Liste_Fichiers

    tell application "Finder"
        display dialog "Nb=" & (count of
Liste_Fichiers) & return & -
                (name of first item of
Liste_Fichiers) giving up after 1
    end tell

end adding folder items to
```

Réalisez ensuite la copie (glissez les quatre fichiers de D1 vers D2 tout en maintenant la touche Alt).

La barre habituelle de progression du Finder apparaît indiquant la copie en cours de quatre fichiers. Soudain, avant la fin de la copie, une fenêtre indiquant « Nb=x » et le nom d'un des fichiers. Cette fenêtre disparaît au bout d'une seconde. Puis sans doute quelques secondes plus tard, de nouveau une fenêtre Nb=y, et ainsi de suite... Vous pouvez avoir deux ou trois fois la fenêtre.

Note : Les sceptiques (la confiance n'exclut pas le contrôle...) peuvent remplacer le « Display dialog » par une commande shell « touch » pour écrire dans un fichier de trace qui n'influe pas sur la vitesse du Finder.

Selon la vitesse et l'ordre de sélection de fichiers (petits d'abord, au milieu ou en dernier), vous aurez sans doute deux ou trois appels du script avec un nombre total de Nb égal à 4 (bien sûr, car il y a quatre fichiers en tout et pas un de plus).

Les quatre séquences suivantes sont possibles avec les nombres pour chaque dialogue : 1-2-1, 1-1-2, 2-1-1, ou 2-2. Si vous avez 1-1-1-1, votre copie est particulièrement lente !

Supprimez les fichiers du dossier D2 et recommencez avec un transfert (D1 et D2 sur le même volume, glissez simplement les fichiers de D1 à D2). Il est fort probable que vous n'aurez qu'une seule action avec Nb=4 ! Tout simplement parce que dans ce cas, le transfert est plus rapide que le délai de latence de l'action de dossier.

J'ai cherché à tester la limite de la temporisation interne.

Un transfert de 1000 fichiers de 4 Ko chacun ne génère

qu'une seule action de dossier avec une liste de 1000 fichiers. Sur une machine rapide (iMac27 quad core i7), la copie de ces fichiers donne le même résultat. Par contre, dès que la liste des fichiers contient un seul fichier de 1 Go au milieu, la copie générera au moins deux actions de dossier.

Donc, règle N° 1 : ne jamais supposer que tous les fichiers ajoutés vous arriveront en même temps dans le script.

Lors du test précédent, les lecteurs les plus observateurs auront sans doute remarqué que la fenêtre du script s'affiche AVANT la copie complète des fichiers en question.

Ce n'est pas important pour de nombreux scripts, mais si vous devez modifier les fichiers ajoutés (voire les supprimer s'ils ne correspondent pas à vos critères) et que, en plus les fichiers sont gros (1, 2 ou 10 Go...), alors là, oui, nous avons un problème, car votre script commencera à travailler non pas « after », mais « pendant ».

Il se peut que vous ayez besoin d'être certain que le fichier est bel et bien ajouté dans le dossier. Dans ce cas, il faut que votre script sache détecter quand la copie est effectivement terminée, pour ensuite se dérouler comme vous l'avez prévu. C'est ce que nous allons voir dans le prochain paragraphe.

## **5.2. Comment forcer le script à bien attendre le « after » de l'instruction ?**

Sur la toile, vous trouverez cette question récurrente (détection de fin de copie), pas seulement pour des actions de dossier, parfois avec des réponses fausses ou approximatives. J'ai testé un certain nombre de méthodes que je vous liste ci-dessous : celles qui ne fonctionnent pas et celles qui fonctionnent.

Ce n'est certainement pas exhaustif, alors n'hésitez pas à partager vos propres méthodes.

### **5.2.1. Les méthodes qui ne fonctionnent pas (pour vous éviter des heures perdues)**

D'après la documentation AppleScript, il devrait y avoir plusieurs possibilités pour déterminer si un fichier est en cours de copie ou pas, s'il existe, s'il est en écriture ou bloqué. Malheureusement beaucoup sont inopérantes, dans le cadre d'une action de dossier :

utiliser la propriété « exist » du fichier : non, car elle est « true » dès que l'entrée est inscrite dans le répertoire... Bien avant la fin de copie ! ;

utiliser la propriété « Busy Status » : non, car, bien que Apple demande aux développeurs de la gérer, le Finder ne la gère pas ! (faites comme je dis, pas comme je fais) ;

utiliser la propriété "File Type of Info for" : curieusement, sa valeur est "brok" (testé sur Tiger et Snow Leopard, Moutain Lion à vérifier) si le fichier est en cours de copie via le Finder. Mais elle est vide lors d'un ajout par une application ou la copie par commande Shell Unix. On ne peut donc s'y fier ;

utiliser la commande « open access with write permission » ne donne pas d'erreur pendant la copie, comme si le fichier n'avait pas de contrainte d'accès. Mais on a parfois une erreur lors de l'écriture d'un bloc. Ouf ! Ça c'est normal, mais on ne veut pas forcément écrire sur le fichier ;

attendre que la fenêtre Finder de progression de copie ne

soit plus affichée : imparfait, car le script peut être face à plusieurs copies simultanées dans le Finder sans savoir quelle fenêtre de copie doit disparaître.

### **5.2.2. Les méthodes qui fonctionnent**

Je n'ai trouvé qu'une méthode sur la toile dont le concept fonctionne vraiment (auteur anonyme, merci à lui). Je l'ai simplement remise en forme ici.

Avant de la trouver, j'en avais créé une qui, j'avoue, n'est pas très élégante, mais j'adore la commande Unix ls ! (elle est souvent beaucoup plus rapide qu'AppleScript et puis cela fait un peu travailler les neurones).

Méthode basée sur la taille du fichier : si la taille du fichier ne change plus, c'est que tout est copié ! C'est la plus simple.

```
set Mon_Fichier to "dossier:fichier" -- mettre
ici le chemin et nom de votre fichier

tell application "Finder"
  try
    set ASize to -1
    repeat until (size of Mon_Fichier) =
ASize
      set ASize to (size of Mon_Fichier)
      delay 0.3
    end repeat
  end try
end tell

display dialog "Le fichier est totalement
copié !"
```

Méthode basée sur les résultats de la commande shell/Unix ls -l. Cette commande affiche, en début de ligne, les autorisations Unix du fichier :

- avant copie -> erreur, car pas de fichier !
- pendant copie Finder -> -rwxr-xr-x@ (@ en position 11 et débute par -rwx)
- pendant copie Unix -> -rwx-----
- après copie : -rwx, mais plus de @ ni de suite "-----". La valeur exacte dépend des autorisations sur le fichier (user, group...).
- Note : j'ai cherché sur le Net, sans succès, la signification du « @ » à la fin des autorisations.

```
set Mon_Fichier to "dossier:fichier" -- mettre
ici le chemin et nom de votre fichier

set F_Unix to quoted form of POSIX path of
Mon_Fichier
set LS_Debut to ""
set PosArobase to "@"
set LS_Fin to "-----"
set T to "xxxxxxxxxxxxxx"
repeat until (LS_Debut = "-rwx") and not
(PosArobase = "@") and not (LS_Fin = "-----")
  try
    set T to do shell script "ls -l " &
F_Unix
  on error
    set T to "xxxxxxxxxxxxxx"
  end try
  set LS_Debut to text 1 thru 4 of T
  set LS_Fin to text 5 thru 10 of T
  set PosArobase to character 11 of T
end repeat
```

```
display dialog "Le fichier est totalement  
copié !"
```

Pour résumer, si vous manipulez de petits fichiers et que vous changez les noms, pas de souci pour votre action de dossier.

Si vous manipulez de gros fichiers et que vous voulez les traiter une fois entièrement copiés (et seulement à ce moment), vous devez insérer l'une de ces méthodes dans votre script.

La méthode la plus « propre », c'est-à-dire lisible, consiste à faire une boucle sur les fichiers de *Liste\_Fichiers* et, dans cette boucle, d'insérer l'un de ces scripts. Votre script continuera ensuite dans la boucle, seulement lorsque la copie du fichier en question sera terminée.

Ainsi chaque fichier sera traité par le script **après** sa copie, pas pendant.

On peut aussi n'utiliser seulement qu'une fois l'un de ces scripts, avant la boucle de traitement sur *Liste\_Fichiers*. Il suffit de vérifier si le dernier élément de cette liste (instruction *last item of Liste\_Fichiers*) a terminé sa copie. Ainsi on est certain que tous les fichiers sont copiés avant de démarrer l'action de dossier.

En revanche, on attend effectivement la dernière copie pour commencer le script. Bien qu'il fonctionne, le risque est qu'une future version du Finder fasse du parallélisme (« multithreading ») à l'intérieur d'une seule action de copie. Alors vous ne pourrez plus savoir si le dernier fichier de la liste est bien celui qui sera traité en dernier.

Les versions actuelles du Finder font du parallélisme entre les tâches de copies, mais pas à l'intérieur d'une tâche de copie multiple. Pour combien de temps encore ?

À vous de voir en fonction de ce que fera le reste de votre script ! En tout cas, vous voilà prévenu.

### 5.3. Exemple pratique sur le dossier de téléchargement

Afin de mettre en pratique le problème de synchronisation des tâches, voici ci-dessous un exemple basé sur l'ajout, via votre navigateur favori, d'un fichier dans le dossier téléchargement. Il est à peu près certain dans ce cas que le script d'action de dossier débutera avant que le fichier soit entièrement téléchargé. Il est tout aussi probable que l'action ne comportera qu'un fichier ajouté.

L'action elle-même copiera le fichier téléchargé dans un sous-dossier en fonction du type de fichier. Si ce sous-dossier n'existe pas, l'action le crée (gestion des erreurs) et le script passe outre ce nouvel ajout.

Il y a aussi une astuce afin de contourner un comportement particulier du navigateur Safari. En effet, celui-ci, lors d'un téléchargement, crée parfois un fichier temporaire qu'il va renommer ensuite, une fois le chargement effectué.

Par exemple, il va déclencher l'action de dossier avec l'ajout d'un fichier *Fichier.Ext.Download*, dont l'extension est « download » puis modifier le nom en *Fichier.Ext* dont l'extension deviendra « Ext ». Curieusement (et heureusement !), la référence alias du fichier reste la même ce qui permet sa copie.

Cependant, il faut bien tenir compte de cela pour assigner ce fichier au dossier adéquat (celui qui doit contenir vos fichiers avec l'extension « Ext »).

Le petit script ci-dessous contourne le problème en copiant immédiatement le nom en mémoire (avant fin de téléchargement) pour en extraire l'extension qui servira à

l'assignation du sous-dossier voulu. Il a été testé avec Safari et plusieurs sites et types de fichiers. Il est suffisamment commenté pour que vous puissiez l'adapter à votre cas particulier.

Bien sûr, ce script est à assigner comme action du dossier téléchargement (celui que vous avez sélectionné dans les préférences de votre navigateur).

```
-- Action de dossier sur le dossier  
Téléchargement :  
-- Transfère les fichiers vers des sous-dossiers  
par type  
-- une fois leur téléchargement terminé  
  
-- Les sous-dossiers regroupent les fichiers par  
groupes d'extensions  
-- Pour chaque groupe sont définis les types et  
le nom du sous-dossier  
property FImage : {"JPG", "jpg", "png", "TIFF"}  
property DImage : "Mes_Images"  
  
property FTexte : {"txt", "doc", "docx", "odt"}  
property DTexte : "Mes_Textes"  
  
property FVideo : {"dv", "DV", "mov", "MOV",  
"VOB", "vob", "divx", "mp4", "MP4"}  
property DVideo : "Mes_Videos"  
  
property FAudio : {"m4a", "M4A", "aif", "mp3",  
"aiff"}  
property DAudio : "Mes_Musiques"  
  
-- Pour toutes les autres extensions (impossible  
de lister tous les types !)  
property DAutres : "Autres formats"  
  
on adding folder items to Mon_Dossier after  
receiving Liste_Fichiers  
    tell application "Finder"  
        -- Définit une liste de tous les sous-  
dossiers possibles  
        set Dossiers_Ajout to {DImage, DTexte,  
DVideo, DAudio, DAutres}  
        -- Boucle sur chaque fichier ajouté dans  
le dossier de téléchargement  
        repeat with Mon_Item in Liste_Fichiers  
            set Mon_Ext to name extension of  
Mon_Item  
  
            set Mon_Nom to name of Mon_Item  
            -- Safari télécharge parfois en  
créant un nom temporaire avec "download"  
            -- Exemple : photo.jpg.download  
            -- Cela déclenche l'action de  
dossier, mais Safari change ensuite le nom du  
fichier!!  
            -- Si Safari change avant que  
l'action de dossier ne démarre, cela crée une  
erreur  
            -- On commence donc par prendre le  
nom sans le ".download" s'il existe  
            -- On extrait l'extension pour  
attribution du dossier  
            if Mon_Ext is "download" then  
                -- On supprime le download (les  
neuf derniers caractères)  
                set Mon_Temp to text 1 thru  
((length of Mon_Nom) - 9) of Mon_Nom  
                -- Recherche du premier "." en  
partant de la fin pour l'extension  
                set I to length of Mon_Temp  
                repeat while character I of  
Mon_Temp is not "."
```

```

        set I to I - 1
    end repeat
    set Mon_Ext to text (I + 1) thru
(length of Mon_Temp) of Mon_Temp
    set Mon_Nom to text 1 thru I of
Mon_Temp
    end if

    -- attente de fin de téléchargement
(vérification que la taille ne change plus)
    try
        set Mon_Fichier to Mon_Item as
alias
        set OldSize to -1
        repeat until (size of
Mon_Fichier) = OldSize
            set OldSize to (size of
Mon_Fichier)
            delay 0.5
        end repeat
    end try
    -- On teste si l'ajout n'est pas
justement l'un des sous-dossiers
    -- Si c'est le cas, la boucle ne fait
rien et passe à l'item suivant !
    if Mon_Nom is not in Dossiers_Ajout
then
        -- Détermination du sous-dossier
via l'extension
        if Mon_Ext is in FImage then
            set Sous_Dossier to DImage
        else if Mon_Ext is in FTexte then
            set Sous_Dossier to DTexte
        else if Mon_Ext is in FVideo then
            set Sous_Dossier to DVideo
        else if Mon_Ext is in FAudio then

```

```

        set Sous_Dossier to DAudio
    else
        set Sous_Dossier to DAutres
    end if

    -- Vérification existence du
sous-dossier, création si nécessaire
    if not (exists folder
Sous_Dossier of Mon_Dossier) then
        make new folder at
Mon_Dossier with properties {name:Sous_Dossier}
        -- attention, cet ajout
génère encore un event add folder items !! ->
d'où le premier test !
    end if

    -- Transfert du fichier vers le
sous-dossier adéquat
    move Mon_Item to folder
Sous_Dossier of Mon_Dossier as alias

    end if -- si l'item n'est pas un
sous-dossier
    end repeat -- fin de boucle sur chaque
item ajouté

    end tell
end adding folder items to

```

J'espère que ces explications et les exemples vous seront utiles pour écrire et gérer des actions de dossiers. Cet outil si puissant que les autres OS n'ont pas... Bon courage !

Retrouvez l'article de Pbell en ligne : [Lien 06](#)



### Évoluer vers une architecture MVC en PHP

L'objectif de cet article est de découvrir comment améliorer l'architecture d'un site Web en passant d'une organisation classique (monopage) à une organisation respectant le modèle MVC.

Il s'agit d'une adaptation d'un cours donné aux étudiants de seconde année de BTS SIO (Services Informatiques aux Organisations) au lycée La Martinière Duchère de Lyon.

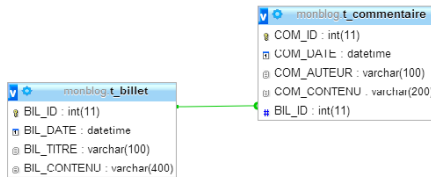
Remarque : cet article s'inspire en partie de la page Web [Symfony2 versus flat PHP Lien 07](#).

#### 1. Présentation du contexte d'exemple

Nous mettrons en œuvre les principes présentés dans cet article sur un exemple simple : une page Web PHP de type blog interagissant avec une base de données relationnelle. Vous trouverez les fichiers sources du contexte initial à l'adresse [Lien 08](#).

##### 1.1. Base de données

La base de données utilisée est très simple. Elle se compose de deux tables, l'une stockant les billets (articles) du blog et l'autre les commentaires associés aux articles.



Cette base de données contient quelques données de test, insérées par le script SQL ci-dessous.

```
INSERT INTO T_BILLET(BIL_DATE, BIL_TITRE, BIL_CONTENU) VALUES (NOW(), 'Premier billet', 'Bonjour monde ! Ceci est le premier billet sur mon blog.');
```

```
INSERT INTO T_BILLET(BIL_DATE, BIL_TITRE, BIL_CONTENU) VALUES (NOW(), 'Au travail', 'Il faut enrichir ce blog dès maintenant.');
```

```
INSERT INTO T_COMMENTAIRE(COM_DATE, COM_AUTEUR, COM_CONTENU, BIL_ID) VALUES (NOW(), 'A. Nonyme', 'Bravo pour ce début', 1);
```

```
INSERT INTO T_COMMENTAIRE(COM_DATE, COM_AUTEUR, COM_CONTENU, BIL_ID) VALUES (NOW(), 'Moi', 'Merci ! Je vais continuer sur ma lancée', 1);
```

##### 1.2. Page principale

Voici le code source PHP de la page principale index.php de notre blog.

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="UTF-8" />
```

```
<link rel="stylesheet" href="style.css" />
<title>Mon Blog - Sans MVC</title>
</head>
<body>
  <div id="global">
    <header>
      <h1 id="titreBlog"><a href="index.php">Mon Blog</a></h1>
      <p>Je vous souhaite la bienvenue sur ce modeste blog.</p>
    </header>
    <nav>
      <section>
        <h1>Billets</h1>
        <ul>
          <li><a href="todo">Billets récents</a></li>
          <li><a href="todo">Tous les billets</a></li>
        </ul>
      </section>
      <section>
        <h1>Administration</h1>
        <ul>
          <li><a href="todo">Écrire un billet</a></li>
        </ul>
      </section>
    </nav>
    <div id="contenu">
      <?php
        $bdd = new PDO(
          'mysql:host=localhost;dbname=monblog;charset=utf8',
          'root', ''
        );
        $requeteBillets = "select * from T_BILLET order by BIL_ID desc";
        $billets = $bdd->query($requeteBillets);
        foreach ($billets as $billet):
          ?>
          <article>
            <header>
              <h1 class="titreBillet">
                <?php
                  $billet['BIL_TITRE'] ?>
                </h1>
              <time><?php
                $billet['BIL_DATE'] ?></time>
```

```

        </header>
        <p><?=
$billet['BIL_CONTENU'] ?></p>
        </article>
        <hr />
        <?php endforeach; ?>
    </div> <!-- #contenu -->
    <footer id="piedBlog">
        Blog réalisé avec PHP, HTML5 et
CSS.
    </footer>
</div> <!-- #global -->
</body>
</html>

```

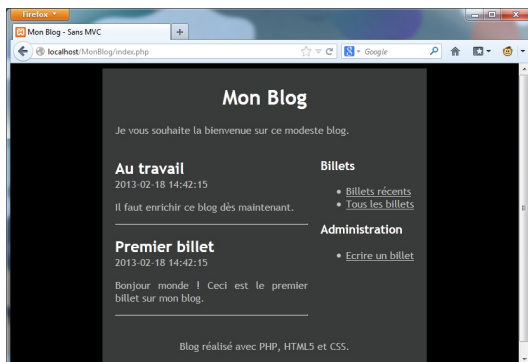
On peut faire les remarques suivantes.

- Cette page est écrite en HTML5 et utilise certaines nouvelles balises, comme <article>.
- Elle présente un menu de navigation (balise <nav>) présent uniquement pour des raisons esthétiques (les liens ne fonctionnent pas).
- Elle emploie l'affichage abrégé <?= ... ?> ([Lien 09](#)) plutôt que <?php echo ... ?>, ainsi que la syntaxe alternative ([Lien 10](#)) pour la boucle foreach.
- Elle utilise l'extension PDO ([Lien 11](#)) de PHP afin d'interagir avec la base de données.

Pour le reste, il s'agit d'un exemple assez classique d'utilisation de PHP pour construire une page dynamique affichée par le navigateur client.

### 1.3. Affichage obtenu

Le résultat obtenu depuis un navigateur client est le suivant.



Ce résultat est dû à l'application d'une feuille de style CSS (fichier style.css) afin d'améliorer le rendu HTML. Pour information, voici le code source associé.

```

/* Pour pouvoir utiliser une hauteur (height) ou
une hauteur minimale
(min-height) sur un bloc, il faut que son
parent direct ait lui-même une
hauteur déterminée (donc toute valeur de
height sauf "auto": hauteur en
pixels, em, autres unités...).
Si la hauteur du parent est en pourcentage,
elle se réfère alors à la
hauteur du « grand-père », et ainsi de suite.
Ainsi, pour pouvoir utiliser un "min-height:
100 %" sur div#global, il nous
faut:
- un parent (body) en "height: 100 %";

```

```

- le parent de body également en "height: 100
%". */
html, body {
    height: 100%;
}

body {
    color: #bfbfbf;
    background: black;
    font-family: 'Futura-Medium', 'Futura',
'Trebuchet MS', sans-serif;
}

a {
    color: #bfbfbf;
}

a:hover, a:focus {
    color: crimson;
}

nav {
    float: right;
    margin-left: 20px;
}

h1 {
    color: white;
}

#global {
    min-height: 100%; /* Voir commentaire sur
html et body en haut de la feuille de style */
    background: #333534;
    width: 60%;
    margin: auto; /* Permet de centrer la div
*/
    text-align: justify;
    padding: 5px 20px;
}

#contenu {
    margin-bottom : 30px;
    overflow: hidden; /* Évite au bloc central
de glisser sous le menu latéral
(alternative à la
définition de marges pour ces blocs) */
}

#titreBlog, #piedBlog {
    text-align: center;
}

.titreBillet {
    margin-bottom : 0px;
}

.titreBillet a, #titreBlog a {
    color: white;
    text-decoration: none; /* Désactive le
soulignement des liens */
}

```

### 1.4. Critique de l'architecture actuelle

Les principaux défauts de cette page Web sont les suivants :

- elle mélange balises HTML et code PHP ;
- sa structure est monobloc, ce qui rend sa réutilisation difficile.

On sait que tout logiciel doit gérer plusieurs problématiques :

- interactions avec l'extérieur, en particulier l'utilisateur : saisie et contrôle de données, affichage. C'est la problématique de **présentation** ;
- opérations sur les données (calculs) en rapport avec les règles métier (« business logic »). C'est la problématique des **traitements** ;
- accès et stockage des informations qu'il manipule, notamment entre deux utilisations. C'est la problématique des **données**.

La page Web actuelle mélange code de présentation (les balises **HTML**) et accès aux données (requête **SQL**). Ceci est totalement contraire au principe de **responsabilité unique**, le principal principe de conception logicielle.

L'architecture actuelle montre ses limites dès que le contexte se complexifie. Le volume de code des pages PHP explose et la maintenance devient délicate. Il faut faire mieux.

## 2. Mise en place d'une architecture MVC simple

### 2.1. Amélioration de l'exemple

#### 2.1.1. Isolation de l'affichage

Une première amélioration consiste à séparer le code d'accès aux données du code de présentation au sein du fichier `index.php`.

```
<?php
$dbdd = new
PDO('mysql:host=localhost;dbname=monblog;charset=
utf8',
    'root', '');
$requeteBillets = "select * from T_BILLET order
by BIL_ID desc";
$billets = $dbdd->query($requeteBillets);
?>

<!doctype html>
<head>
...
    <div id="contenu">
        <?php
            foreach ($billets as $billet):
                ?>
                    <article>
                        <header>
                            <h1
class="titreBillet">
                                <?=$
                                $billet['BIL_TITRE'] ?>
                                    </h1>
                                    <time><?=$
                                $billet['BIL_DATE'] ?></time>
                            </header>
                            <p><?=$
                                $billet['BIL_CONTENU'] ?></p>
                        </article>
                    <hr />
                <?php endforeach; ?>
            </div> <!-- #contenu -->
...

```

Le code est devenu plus lisible, mais les problématiques de présentation et d'accès aux données sont toujours gérées au sein d'un même fichier PHP. En plus de limiter la modularité, ceci est contraire aux bonnes pratiques de développement PHP (norme PSR-1 : [Lien 12](#)).

On peut aller plus loin dans le découplage en regroupant le code d'affichage précédent dans un fichier dédié nommé `listeBillets.php`.

```
<!doctype html>
<html lang="fr">
<head>
...
</head>
<body>
...
    <div id="contenu">
        <?php
            foreach ($billets as $billet):
                ?>
                    <article>
                        ...
                    </article>
                <hr />
            <?php endforeach; ?>
    </div> <!-- #contenu -->
...
</body>
</html>

```

La page principale `index.php` devient alors :

```
<?php
// Accès aux données
$dbdd = new
PDO('mysql:host=localhost;dbname=monblog;charset=
utf8',
    'root', '');
$requeteBillets = "select * from T_BILLET order
by BIL_ID desc";
$stmtBillets = $dbdd->query($requeteBillets);

// Affichage
require 'listeBillets.php';

```

Rappel : la fonction PHP `require` fonctionne de manière similaire à `include` : elle inclut et interprète le fichier spécifié. En cas d'échec, `include` ne produit qu'un avertissement alors que `require` stoppe le script.

La balise de fin de code PHP `?>` est volontairement omise à la fin du fichier `index.php`. C'est une bonne pratique ([Lien 13](#)) pour les fichiers qui ne contiennent que du PHP. Elle permet d'éviter des problèmes lors d'inclusions de fichiers.

#### 2.1.2. Isolation de l'accès aux données

Nous avons amélioré l'architecture de notre page, mais nous pourrions gagner en modularité en isolant le code d'accès aux données dans un fichier PHP dédié. Appelons ce fichier `modele.php`.

```

<?php

function getBillets()
{
    $bdd = new
PDO('mysql:host=localhost;dbname=monblog;charset=
utf8',
        'root', '');
    $requeteBillets = "select * from T_BILLET
order by BIL_ID desc";
    $stmtBillets = $bdd->query($requeteBillets);
    return $stmtBillets;
}

```

Dans ce fichier, nous avons déplacé la récupération des billets du blog à l'intérieur d'une **fonction** nommée `getBillets`.

Le code d'affichage (fichier `listeBillets.php`) ne change pas. Le lien entre accès aux données et présentation est effectué par le fichier principal `index.php`. Ce fichier est maintenant très simple.

```

<?php

require 'modele.php';

$billets = getBillets();

require 'listeBillets.php';

```

### 2.1.3. Bilan provisoire

Outre la feuille de style CSS, notre page Web est maintenant constituée de trois fichiers :

- `modele.php` (PHP uniquement) pour l'accès aux données ;
- `listeBillets.php` (PHP et HTML) pour l'affichage des billets du blog ;
- `index.php` (PHP uniquement) pour faire le lien entre les deux pages précédentes.

Cette nouvelle structure est plus complexe, mais les responsabilités de chaque partie sont maintenant claires.

En faisant ce travail de **refactoring**, nous avons rendu notre exemple conforme à un modèle d'architecture très employé sur le Web : le modèle MVC.

## 2.2. Le modèle MVC

### 2.2.1. Présentation

Le modèle MVC décrit une manière d'architecturer une application informatique en la décomposant en trois sous-parties :

- la partie **Modèle** ;
- la partie **Vue** ;
- la partie **Contrôleur**.

Ce modèle a été imaginé à la fin des années 1970 pour le langage Smalltalk afin de bien séparer le code de l'interface graphique de la logique applicative. Il est utilisé dans de très nombreux langages : bibliothèques Swing et Model 2 (JSP) de Java, *frameworks* PHP, ASP.NET MVC, etc.

### 2.2.2. Rôles des composants

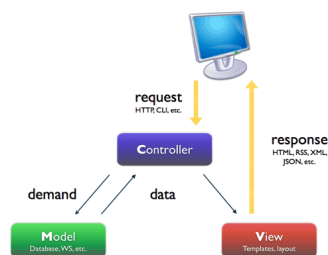
La partie **Modèle** d'une architecture MVC encapsule la logique métier (« *business logic* ») ainsi que l'accès aux données. Il peut s'agir d'un ensemble de fonctions (Modèle procédural) ou de classes (Modèle orienté objet).

La partie **Vue** s'occupe des interactions avec l'utilisateur : présentation des informations, saisie et validation des données.

La partie **Contrôleur** gère la dynamique de l'application. Elle fait le lien entre l'utilisateur et le reste de l'application.

### 2.2.3. Interactions entre les composants

Le diagramme ci-dessous résume les relations entre les composants d'une architecture MVC.



Extrait de la documentation du framework Symfony

- La demande de l'utilisateur (par exemple une requête HTTP) est reçue et interprétée par le **Contrôleur**.
- Celui-ci utilise les services du **Modèle** afin de préparer les données à afficher.
- Ensuite, le **Contrôleur** fournit ces données à la **Vue**, qui les présente à l'utilisateur (par exemple sous la forme d'une page HTML).

Une application construite sur le principe du MVC se compose toujours de trois parties distinctes. Cependant, il est fréquent que chaque partie soit elle-même décomposée en plusieurs éléments. On peut ainsi trouver plusieurs modèles, plusieurs vues ou plusieurs contrôleurs à l'intérieur d'une application MVC.

### 2.2.4. Avantages et inconvénients

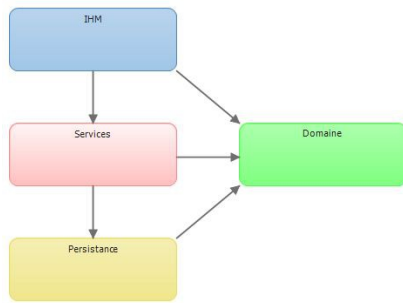
Le modèle MVC offre une séparation claire des responsabilités au sein d'une application, en conformité avec les principes de conception déjà étudiés : responsabilité unique, couplage **faible** et cohésion **forte**. Le prix à payer est une augmentation de la complexité de l'architecture.

Dans le cas d'une application Web, l'application du modèle MVC permet aux pages HTML (qui constituent la **Vue**) de contenir le moins possible de code serveur, étant donné que le *scripting* est regroupé dans les deux autres parties de l'application.

### 2.2.5. Différences avec un modèle en couches

Attention à ne pas employer le terme de « couche » à propos du modèle MVC. Un modèle en couches se caractérise par l'interdiction pour une couche non transversale de communiquer au-delà des couches adjacentes. De plus, le nombre de couches n'est pas imposé : il est fonction de la complexité du contexte.





### 2.3. Améliorations supplémentaires

Même si notre architecture a déjà été nettement améliorée, il est possible d'aller encore plus loin.

#### 2.3.1. Factorisation des éléments d'affichage communs

Un site Web se réduit rarement à une seule page. Il serait donc souhaitable de définir à un seul endroit les éléments communs des pages HTML affichées à l'utilisateur (les vues).

Une première solution consiste à inclure les éléments communs avec des fonctions PHP include. Il existe une autre technique, plus souple, que nous allons mettre en œuvre : l'utilisation d'un modèle de page (gabarit), appelé *template* en anglais. Ce modèle contiendra tous les éléments communs et permettra d'ajouter les éléments spécifiques à chaque vue. On peut écrire ce *template* de la manière suivante (fichier gabarit.php).

```

<!doctype html>
<html lang="fr">
  <head>
    <meta charset="UTF-8" />
    <link rel="stylesheet" href="style.css" />
    <title>= $titre ?&gt;&lt;/title&gt; &lt;!-- Élément
    spécifique --&gt;
  &lt;/head&gt;
  &lt;body&gt;
    &lt;div id="global"&gt;
      &lt;header&gt;
        &lt;h1 id="titreBlog"&gt;Mon Blog&lt;/h1&gt;
        &lt;p&gt;Je vous souhaite la bienvenue sur ce
        modeste blog.&lt;/p&gt;
      &lt;/header&gt;
      &lt;nav&gt;
        &lt;section&gt;
          &lt;h2&gt;Billets&lt;/h2&gt;
          &lt;ul&gt;
            &lt;li&gt;&lt;a href="todo"&gt;Billets
            récents&lt;/a&gt;&lt;/li&gt;
            &lt;li&gt;&lt;a href="todo"&gt;Tous les
            billets&lt;/a&gt;&lt;/li&gt;
          &lt;/ul&gt;
        &lt;/section&gt;
        &lt;section&gt;
          &lt;h2&gt;Administration&lt;/h2&gt;
          &lt;ul&gt;
            &lt;li&gt;&lt;a href="todo"&gt;Écrire un
            billet&lt;/a&gt;&lt;/li&gt;
          &lt;/ul&gt;
        &lt;/section&gt;
      &lt;/nav&gt;
      &lt;div id="contenu"&gt;
        &lt;?= $contenu ?&gt; &lt;!-- Élément spécifique
        --&gt;
    &lt;/div&gt;
  &lt;/body&gt;
&lt;/html&gt;
  </pre

```

```

</div> <!-- #contenu -->
<footer id="piedBlog">
  Blog réalisé avec PHP, HTML5 et CSS.
</footer>
</div> <!-- #global -->
</body>
</html>
  
```

Au moment de l'affichage d'une vue HTML, il suffit de définir les valeurs des éléments spécifiques, puis de déclencher le rendu de notre gabarit. Pour cela, on utilise des fonctions PHP qui manipulent le flux de sortie de la page. Voici notre page listeBillets.php réécrite :

```

<?php $titre = 'Mon Blog ? MVC simple' ?>
<?php ob_start() ?>
<?php foreach ($billets as $billet):
  ?>
  <article>
    <header>
      <h1 class="titreBillet"><?=
      $billet['BIL_TITRE'] ?></h1>
      <time><?= $billet['BIL_DATE'] ?></time>
    </header>
    <p><?= $billet['BIL_CONTENU'] ?></p>
  </article>
  <hr />
<?php endforeach; ?>
<?php $contenu = ob_get_clean() ?>
<?php require 'gabarit.php' ?>
  
```

Ce code mérite quelques explications :

- la première ligne définit la valeur de l'élément spécifique \$titre ;
- la deuxième ligne utilise la fonction PHP ob\_start ([Lien 14](#)). Son rôle est de déclencher la mise en tampon du flux HTML de sortie : au lieu d'être envoyé au navigateur, ce flux est stocké en mémoire ;
- la suite du code (boucle foreach) génère les balises HTML <article> associées aux billets du blog. Le flux HTML créé est mis en tampon ;
- une fois la boucle terminée, la fonction PHP ob\_get\_clean ([Lien 15](#)) permet de récupérer dans une variable le flux de sortie mis en tampon depuis l'appel à ob\_start. La variable se nomme ici \$contenu, ce qui permet de définir l'élément spécifique associé ;
- enfin, on déclenche le rendu du gabarit. Lors du rendu, les valeurs des éléments spécifiques \$titre et \$contenu seront insérées dans le résultat HTML envoyé au navigateur.

L'affichage utilisateur est strictement le même qu'avant l'utilisation d'un gabarit. Cependant, nous disposons maintenant d'une solution souple pour créer plusieurs vues tout en centralisant la définition de leurs éléments communs.

#### 2.3.2. Factorisation de la connexion à la base

On peut améliorer l'architecture de la partie Modèle en isolant le code qui établit la connexion à la base de données sous la forme d'une fonction getBdd ajoutée dans

le fichier `modele.php`. Cela évitera de dupliquer le code de connexion lorsque nous ajouterons d'autres fonctions au **Modèle**.

```
<?php

// Renvoie la liste de tous les billets, triés
par identifiant décroissant
function getBillets() {
    $bdd = getBdd();
    $requeteBillets = "select * from T_BILLET
order by BIL_ID desc";
    $stmtBillets = $bdd->query($requeteBillets);
    return $stmtBillets;
}

// Effectue la connexion à la BDD
// Instancie et renvoie l'objet PDO associé
function getBdd() {
    $bdd = new
PDO('mysql:host=localhost;dbname=monblog;charset=
utf8',
        'root', '');
    return $bdd;
}
```

### 2.3.3. Gestion des erreurs

Par souci de simplification, nous avons mis de côté la problématique de la gestion des erreurs. Il est temps de s'y intéresser.

Pour commencer, il faut décider quelle partie de l'application aura la responsabilité de traiter les erreurs qui pourraient apparaître lors de l'exécution. Ce pourrait être le **Modèle**, mais il ne pourra pas les gérer correctement à lui seul, ni informer l'utilisateur. La **Vue**, dédiée à la présentation, n'a pas à s'occuper de ce genre de problématique. Le meilleur choix est donc d'implémenter la gestion des erreurs au niveau du **Contrôleur**. Gérer la dynamique de l'application, y compris dans les cas dégradés, fait partie de ses responsabilités.

Nous allons tout d'abord modifier la connexion à la base de données afin que les éventuelles erreurs soient signalées sous la forme d'exceptions.

```
...
$bdd = new
PDO('mysql:host=localhost;dbname=monblog;charset=
utf8',
    'root', '',
    array(PDO::ATTR_ERRMODE =>
PDO::ERRMODE_EXCEPTION));
...
```

On peut ensuite ajouter à notre page une gestion minimaliste des erreurs de la manière suivante.

```
<?php

require 'modele.php';

try {
    $billets = getBillets();
    require 'listeBillets.php';
}

catch (Exception $e) {
    echo '<html><body>Erreur ! ' . $e-
```

```
>getMessage() . '</body></html>';
}
```

Le premier `require` inclut uniquement la définition d'une fonction et est placé en dehors du bloc `try`. Le reste du code est placé à l'intérieur de ce bloc. Si une exception est levée lors de son exécution, une page HTML minimale contenant le message d'erreur est affichée.

On peut souhaiter conserver l'affichage du gabarit des vues même en cas d'erreur. Il suffit de définir une vue erreur.php dédiée à leur affichage.

```
<?php $titre = 'Mon Blog ? Erreur !' ?>

<?php ob_start() ?>
<p>Une erreur est survenue : <?= $msgErreur ?
></p>
<?php $contenu = ob_get_clean() ?>

<?php require 'gabarit.php' ?>
```

On modifie ensuite le contrôleur (fichier `index.php`) pour déclencher le rendu de cette vue en cas d'erreur.

```
<?php

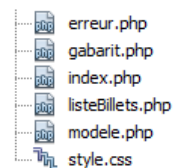
require 'modele.php';

try {
    $billets = getBillets();
    require 'listeBillets.php';
}

catch (Exception $e) {
    $msgErreur = $e->getMessage();
    require 'erreur.php';
}
```

Remarque : dans un contexte professionnel, une gestion d'erreurs efficace impliquerait une journalisation dans un fichier et l'affichage d'un message d'erreur adapté à l'utilisateur.

### 2.4. Bilan : architecture obtenue



Nous avons accompli sur notre page d'exemple un important travail de *refactoring* qui a modifié son architecture en profondeur. Notre page respecte à présent un modèle MVC simple.

L'ajout de nouvelles pages se fait à présent en trois étapes :

- écriture des fonctions d'accès aux données dans le **modèle** ;
- création d'une nouvelle **vue** utilisant le gabarit pour afficher les données ;
- ajout d'une page **contrôleur** pour lier le modèle et la vue.

Retrouvez la suite de l'article de Baptiste Pesquet en ligne : [Lien 16](#).

# Développement Web

## Les derniers tutoriels et articles

### Créer des sélecteurs jQuery personnalisés

Dans cet article, nous allons voir comment adapter jQuery à nos besoins en créant nos propres sélecteurs.

#### 1. Les sélecteurs

Le nom de jQuery vient de « JavaScript Query », ce qui signifie que l'objectif premier de la bibliothèque est la possibilité de récupérer facilement et efficacement des éléments du DOM pour les manipuler. Pour cela, jQuery propose un nombre important de sélecteurs reprenant la syntaxe CSS. Nous ne détaillerons pas ces sélecteurs, mais vous pouvez les retrouver dans l'article Introduction à la bibliothèque JavaScript jQuery ([Lien 17](#)).

La syntaxe CSS permet de cibler des éléments ou des groupes d'éléments de façon très précise et satisfait la plupart des besoins. Cependant, certains manques spécifiques au développement JavaScript ont été décelés par les créateurs de jQuery qui ont donc rajouté des sélecteurs particuliers, notamment pour la gestion des éléments de formulaires. Mais là encore, ils n'ont pas pu penser à tous les cas possibles et vous aurez peut-être des développements spécifiques pour lesquels les sélecteurs existants ne seront pas disponibles.

Vous serez donc parfois amenés à avoir besoin de créer des sélecteurs personnalisés et vous pourrez constater que les développeurs de jQuery ont tout prévu pour vous faciliter la tâche.

#### 2. Créer un sélecteur simple

Créer son propre sélecteur est relativement simple comme vous allez le voir, mais cela implique d'utiliser des fonctions internes de jQuery qui ne sont donc pas renseignées dans la documentation officielle : [Lien 18](#).

Il va falloir rajouter une valeur à l'objet interne `expr` et plus précisément à sa propriété `!`.

Faites bien attention à ce que vous faites. Dans la mesure où vous allez surcharger un objet interne de la bibliothèque, vous devez au préalable vous assurer que vous n'écraserez pas des données existantes ce qui risquerait d'empêcher le bon fonctionnement de jQuery.

La première chose est donc de créer une nouvelle propriété dont le nom sera celui du sélecteur et de lui affecter une fonction :

```
$.expr['!'].monSélecteur = function(){
    // ...
};
```

La valeur « `!` » est tout à fait correcte comme nom de propriété d'un objet, mais elle impose l'utilisation de la notation à crochets car la notation pointée (`expr.!`) provoquerait une erreur.

Cette fonction va itérer parmi tous les éléments possibles

et devra retourner pour chacun d'eux `true` si l'élément doit être conservé, `false` sinon.

Elle prend quatre paramètres automatiques :

- `elem` : l'élément DOM en cours de traitement ;
- `rang` : le compteur pour l'itération en cours ;
- `donnees` : un ensemble d'informations que nous détaillerons plus tard ;
- `collection` : la collection que nous sommes en train de parcourir.

Pour l'instant, seuls les deux premiers paramètres peuvent nous être utiles. Voici comment les utiliser avec deux exemples simples.

```
// Exemple 1 : récupérer tous les éléments
// positionnés (propriété CSS position différente de
// static)
$.expr[':'].positionned = function(elem){
    return $(elem).css('position') != 'static';
};

// Exemple 2 : récupérer les éléments de rang
// pair
$.expr[':'].pair = function(elem, index){
    return index%2;
};
```

Maintenant que vous avez créé votre sélecteur, il ne vous reste plus qu'à l'utiliser dans vos scripts :

```
$(':positionned').css('box-shadow', '0 0 10px 2px
gold');
$('#footer :pair').css('box-shadow', '0 0 10px
2px red');
```

Attention aux performances lorsque vous combinez des sélecteurs de type CSS avec des sélecteurs personnalisés comme dans le second exemple, nous y reviendrons plus tard.

#### 3. Ajouter des paramètres à votre sélecteur

Vous avez probablement déjà rencontré des sélecteurs jQuery prenant des paramètres (par exemple `:eq()` ou `:has()`). Là encore, il vous est tout à fait possible de reproduire cela avec vos paramètres personnalisés.

Pour gérer les paramètres passés au sélecteur, nous allons utiliser le paramètre `donnees` de la fonction de rappel.

Ce paramètre correspond à un tableau contenant quatre données :

- le sélecteur complet ;
- le nom du sélecteur ;
- le type de *quotes* utilisé ;
- les paramètres du sélecteur.

L'exemple suivant va vous montrer à quoi correspondent concrètement ces éléments.

```
$.expr[':'].testParams = function(elem, index,
donnees) {
    alert(donnes.join(' / '));
};
```

Que nous allons appliquer à l'élément HTML suivant :

```
<div id="testSelecteur" style="background-color:
silver; height: 2em; margin: 5px 0;">Élément créé
pour tester les paramètres du sélecteur.</div>
```

Nous avons rajouté `join(' / ')` dans le message d'alerte afin de ne pas confondre la chaîne "parametre1, parametre2" du second exemple avec deux valeurs distinctes du tableau.

Le sélecteur peut être appelé de trois façons différentes : `$(':testParams')`, `$(':testParams()')` et `$(':testParams(paramètres)')`. Les valeurs respectives du dernier paramètre seront : `undefined`, `"` (chaîne vide) et paramètres. Attention à bien prendre en compte que les deux premières possibilités sont supposées être équivalentes.

Nous allons donc pouvoir utiliser l'indice 3 de ce tableau pour gérer nos paramètres.

Nous allons créer un sélecteur qui récupérera les éléments possédant un attribut HTML5 `data-*`.

Pour rappel, HTML5 a introduit ce type d'attribut afin de pouvoir y stocker des données personnalisées manipulables en JavaScript *via* l'objet `dataset`.

Les développeurs de jQuery ont rendu l'objet `data()` compatible avec cette API tout en facilitant son utilisation et nous allons nous en servir pour notre sélecteur. Si aucun paramètre n'est passé, on recherchera tous les éléments ayant un attribut de type `data-*`, sinon, nous ne rechercherons que ceux ayant au moins un attribut correspondant à ceux passés en paramètre. Notre sélecteur se nommera `hasData`.

```
$.expr[':'].hasData = function(elem, index, noms)
{
    // On commence par stocker l'objet data de
l'élément en cours dans une variable
    $dataElem = $(elem).data();
    // Si l'objet est vide, on n'accepte pas
l'élément
    if($.isEmptyObject($dataElem)){
        return false;
    }
    // Sinon, si aucun paramètre n'est présent,
on peut accepter l'élément
    if(!noms[3]){
        return true;
    }
    // Sinon, il faut séparer les paramètres et
vérifier la présence pour chacun d'eux
    var tabNoms = noms[3].replace(/\s/g,
'').split(',');
    var retour = false;
    $.each(tabNoms, function(i, nom){
        if($dataElem[nom] !== undefined){
            // Si l'attribut existe bien, on peut
accepter l'élément
            retour = true;
        }
    });
}
```

```
// Inutile de continuer la boucle
return false;
    }
});
return retour;
};
```

Et voilà, nous avons désormais un sélecteur personnalisé acceptant des paramètres.

#### **4. Aller plus loin : utilisation des sélecteurs personnalisés**

Il existe quelques précautions à prendre pour optimiser le code utilisant des sélecteurs personnalisés (ou des sélecteurs non standard de jQuery).

En effet, pour améliorer les performances liées à la sélection d'éléments, jQuery utilise en interne des méthodes du DOM pour récupérer les éléments, notamment, des méthodes performantes comme `getElementById()` et `querySelectorAll()`. Or si vous utilisez des sélecteurs combinés contenant une syntaxe non compatible avec ces méthodes du DOM, jQuery va être obligé de parcourir tous les éléments concernés par la sélection pour déterminer lesquels garder.

Cela ne signifie pas qu'il ne faut pas utiliser les sélecteurs personnalisés, mais juste qu'il faut essayer autant que possible de séparer leur utilisation de celles des sélecteurs compatibles CSS.

À titre d'exemple, si vous souhaitez récupérer les éléments ayant un attribut `data-*` enfants d'un élément possédant un identifiant donné, évitez la syntaxe

```
$('#element :hasData');
```

mais préférez séparer la recherche de l'élément possédant l'identifiant de celle des éléments possédant l'attribut :

```
$('#element').find(':hasData');
```

Plus la page contiendra d'éléments, plus le gain de performances sera important.

Cette règle s'applique bien entendu à vos sélecteurs personnalisés, mais aussi aux sélecteurs natifs de jQuery ne correspondant pas aux standards CSS (globalement, tous ceux commençant par `:`).

Notez que cette règle s'applique aussi dans d'autres cas. Si votre sélecteur doit d'abord rechercher un identifiant (quelle que soit la suite de la recherche), il est recommandé de séparer la récupération de cet élément du reste du sélecteur.

Dans cette optique, il est important de se souvenir qu'un second paramètre peut être passé à la fonction `$()` qui correspond au contexte auquel se réfère la recherche. Ainsi, des sélecteurs tels que `$('#element :hasData')` ou `$('#element span')` seraient avantageusement remplacés par `$(':hasData', '#element')` ou `$('span', '#element')`.

#### **5. Conclusion**

La création de sélecteurs personnalisés est particulièrement simple si l'on connaît les mécanismes internes de jQuery. Néanmoins, retenez bien ces deux



points importants lorsque vous créez vos nouveaux sélecteurs :

- attention à ne pas écraser des sélecteurs existants, pour cela, vous pouvez commencer votre code par une condition telle que `if(!('selecteur' in $.expr[':'])) {...} ;`

- souvenez-vous que l'utilisation d'un sélecteur personnalisé implique de parcourir tous les éléments pour déterminer lesquels correspondent, ciblez donc convenablement votre recherche avant d'utiliser vos propres sélecteurs.

*Retrouvez l'article de Didier Mouronval en ligne : [Lien 19](#)*

## Les derniers tutoriels et articles

### Développement Web avec Maven Tomcat et Jetty

Cet article fait partie d'une série d'articles autour de l'industrialisation du développement Java avec Maven. Dans cette série je m'attarderai sur les astuces et bonnes pratiques permettant de gagner du temps et de standardiser vos développements. De plus, tout le code de l'article ainsi que l'article lui-même sont disponibles sous GitHub pour permettre à chacun de contribuer à tenir ces articles à jour.

Pour cet article je vous propose de faire le point sur les plugins permettant de jouer avec Tomcat 7.x et Jetty 8.x. Ces plugins très pratiques vous permettront :

- de démarrer un Tomcat/Jetty sans effort en phase de développement ;
- de configurer vos logs ;
- de paramétrer une datasource JNDI pour votre conteneur et d'initialiser votre schéma ;
- de déboguer une application Web dans votre IDE préféré ;
- de compiler vos JSP avant déploiement.

Et voici les plugins qui vont nous intéresser :

- maven-tomcat-plugin
- maven-jetty-plugin
- maven-jspc-plugin pour Jetty et pour Tomcat
- maven-sql-plugin
- maven-failsafe

#### 1. Introduction

Tout d'abord, une première question qui peut vous venir à l'esprit :

Mais pourquoi ces plugins plutôt qu'un simple Tomcat installé sur un répertoire à part ?

L'objectif de ces plugins est multiple :

- accélérer votre cycle de développement ;
- centraliser dans votre code source une configuration qui marche.

En alternative vous auriez pu :

- installer Tomcat ;
- préconfigurer Tomcat pour accueillir votre application ;
- copier le fichier war généré dans le répertoire webapps.

Cette stratégie est fastidieuse, interrompt votre cycle de développement et nécessite que vous préconfiguriez votre conteneur Tomcat de la même façon pour tout le monde dans l'équipe.

Une variation de la méthode ci-dessus aurait été de faire pointer un fichier de contexte d'un Tomcat préinstallé sur vos sources. Cela évite la copie du fichier, mais pas la nécessité d'avoir un Tomcat préconfiguré et uniforme sur toutes les machines.

Une autre variante consiste à utiliser le plugin WTP

d'Eclipse. Là encore, il y a un effort de configuration à refaire sur chaque poste.

Ici nous allons insister sur un principe simple et pourtant fondamental en informatique : DRY (*Don't Repeat Yourself* : [Lien 20](#)). Si vous devez reconfigurer votre environnement de travail pour chaque nouvel arrivant, c'est de la perte de temps.

*Checkout and Run* : notre objectif c'est qu'un nouvel arrivant sur un projet n'ait qu'à cloner le repository et puisse être tout de suite opérationnel.

Si on souhaite accélérer son cycle de développement et éviter les redémarrages trop fréquents, on pourrait aussi utiliser Jrebel : [Lien 21](#). Il existe d'ailleurs un plugin Maven pour créer votre fichier rebel.xml à partir de votre POM : [Lien 22](#). Cette approche ne s'oppose pas à celle que nous allons détailler ici.

#### 2. Configuration de base

##### 2.1. Tomcat

Commençons par la configuration de base de cet article. Démarrons par le projet 1.developpez-webapp : [Lien 23](#). Le seul point d'attention pour l'instant c'est que nous allons déclarer le plugin Tomcat dans le fichier pom.xml dans la section build -> pluginManagement :

```
<plugin>

<groupId>org.apache.tomcat.maven</groupId>
  <artifactId>tomcat7-maven-
plugin</artifactId>
  <version>2.0</version>
</plugin>
```

Ensuite, il vous suffit de lancer la commande suivante :

```
mvn tomcat7:run
```

Vous devriez obtenir le résultat suivant :

```
[INFO] Running war on http://localhost:8080/developpez-webapp
[INFO] Creating Tomcat server configuration at D:\developpez\checkout\developpez\maven-tomcat\developpez-webapp\target\tomcat
Mar 10, 2013 2:35:58 PM org.apache.catalina.startup.Embedded start
Info: Starting tomcat server
Mar 10, 2013 2:35:58 PM org.apache.catalina.core.StandardEngine start
Info: Starting Servlet Engine: Apache Tomcat/6.0.29
Mar 10, 2013 2:35:58 PM org.apache.coyote.http11.Http11Protocol init
Info: Initializing Coyote HTTP/1.1 on http://8080
Mar 10, 2013 2:35:58 PM org.apache.coyote.http11.Http11Protocol start
Info: Starting Coyote HTTP/1.1 on http://8080
```

La commande **mvn tomcat7:run** invoque la phase mvn compile avant de s'exécuter elle-même. Cependant vous profitez de la compilation incrémentale de Maven et vous ne recompilerez pas l'ensemble du projet à chaque fois.

Pour résumer, le conteneur Tomcat est automatiquement téléchargé puis démarré avec le contenu de votre application Web. Le port par défaut est 8080 et le contexte correspond au nom de votre artefact. Vous pouvez donc vous connecter sur :

<http://localhost:8080/developpez-webapp/>

Ici rien de magique, l'application est juste constituée d'un fichier HTML statique.

## 2.2. Jetty

Si vous préférez Jetty, la configuration pour lancer votre conteneur préféré sera sensiblement très proche :

```
<plugin>

<groupId>org.mortbay.jetty</groupId>
  <artifactId>jetty-maven-
plugin</artifactId>
  <version>8.1.5.v20120716</version>
</plugin>
```

puis la commande :

```
mvn jetty:run
```

Par défaut le plugin démarre votre application sur le contexte « / ». L'URL pour y accéder est donc :

<http://localhost:8080>

On remarquera pour le troll que le démarrage avec Jetty est légèrement plus rapide.

## 3. Les logs

Un des aspects les plus importants d'une application c'est qu'elle nous dise ce qu'elle fait.

Disons que vous ayez vos habitudes avec logback ([Lien 24](#)) et que vous souhaitiez l'utiliser ici.

Pour que ce soit intéressant, nous allons ajouter un peu de code sinon par défaut nous n'aurions que les logs de démarrage de Tomcat/Jetty, ce qui diminue un peu l'intérêt de la démonstration. Nous allons donc ajouter un service REST ([Lien 25](#)) construit avec JAX-RS. Ce code va faire appel à plusieurs bibliothèques tierces : Spring, Jackson,

CXF, etc. et chacune de ces bibliothèques va produire ses propres logs.

Note : pour créer mon « Hello World » j'ai utilisé la commande **mvn archetype:generate**, une autre commande très utile pour démarrer avec un bon squelette d'application.

Le code est présent dans le répertoire **2.developpez-webapp-jaxrs** ([Lien 26](#)) et nous ne décrivons pas l'application JAX-RS.

Rajoutons les dépendances pour les logs :

```
<!-- Logging -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.2</version>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-
slf4j</artifactId>
  <version>1.7.2</version>
</dependency>
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-
classic</artifactId>
  <version>1.0.9</version>
</dependency>
<dependency>
  <groupId>ch.qos.logback</groupId>
  <artifactId>logback-core</artifactId>
  <version>1.0.9</version>
</dependency>
```

Si on démarre avec **mvn tomcat7:run** ou **mvn jetty:run**, on remarque que notre application devient très bavarde. Cependant nous avons tous nos logs en DEBUG ce qui n'est pas toujours pratique.

Ajoutons une propriété système à utiliser au démarrage pour nos plugins Jetty et Tomcat afin de configurer les logs avec un fichier de configuration.

Pour le plugin Tomcat :

```
<configuration>
  <systemProperties>
<logback.configurationFile>src/test/resources/log
back.xml</logback.configurationFile>
  </systemProperties>
</configuration>
```

Pour le plugin Jetty c'est un tout petit peu plus verbeux :

```
<configuration>
  <systemProperties>
    <systemProperty>
<name>logback.configurationFile</name>
<value>src/test/resources/logback.xml</value>
    </systemProperty>
  </systemProperties>
</configuration>
```

avec notre fichier de configuration pour logback

(Lien 27) :

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration scan="true" scanPeriod="10
seconds">

  <appender name="STDOUT"
class="ch.qos.logback.core.ConsoleAppender">
    <!-- encoders are assigned the type
ch.qos.logback.classic.encoder.PatternLayoutEncod
er by default -->
    <encoder>
      <pattern>%d{HH:mm:ss.SSS} [%thread]
%-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>

  <root level="info">
    <appender-ref ref="STDOUT" />
  </root>
</configuration>
```

Et voilà, désormais notre application est en mode INFO. Petit bonus, vous remarquerez que nous avons configuré le rechargement automatique du fichier de log avec l'attribut scan=true dans le fichier précédent. Vous pouvez donc changer la configuration de vos logs à la volée pendant que l'application tourne.

#### 4. Configurer une datasource

Un autre besoin qui revient régulièrement, c'est celui de pouvoir configurer une datasource sur le conteneur sous forme de ressources JNDI. Ici nous allons utiliser plusieurs astuces :

- une datasource avec une base de données embarquée H2 ;
- l'utilisation du plugin maven-sql pour initialiser notre schéma (\*).

Le code se trouve dans le répertoire **3.developpez-webapp-jndi** : [Lien 28](#).

(\*) Si vous utilisez JPA avec Hibernate vous pouvez aussi créer votre schéma automatiquement via la directive create-schema. Nous ne verrons pas ce point ici ce qui nous permet de continuer à illustrer des plugins Maven.

##### 4.1. Tomcat

Tout d'abord nous allons configurer le plugin Maven pour utiliser un fichier context.xml custom :

```
<configuration>
<contextFile>src/test/resources/context.xml</cont
extFile>
</configuration>
```

Le fichier context.xml ([Lien 29](#)) :

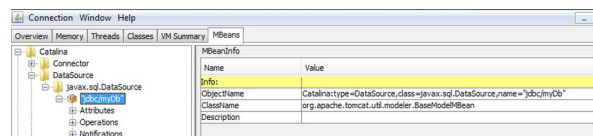
```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
  <Resource name="jdbc/myDb" auth="Container"
type="javax.sql.DataSource"
driverClassName="org.h2.Driver"
```

```
url="jdbc:h2:target/data;MODE=PostgreSQL;DB_CLOSE
_DELAY=-1"
username="sa" password="sa"
maxActive="10" maxIdle="2"
/>
  <ResourceLink global="jdbc/myDb"
name="jdbc/myDb" type="javax.sql.DataSource" />
</Context>
```

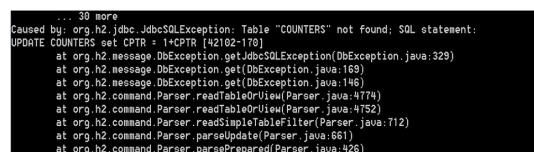
Et dernier point, votre conteneur doit désormais connaître le driver de votre base de données. Dans un cas classique, vous poseriez votre jar dans le répertoire lib de Tomcat. Ici vous allez utiliser le tag dependency lié au plugin :

```
<plugin>
<groupId>org.apache.tomcat.maven</groupId>
<artifactId>tomcat7-maven-
plugin</artifactId>
<version>2.0</version>
<dependencies>
<dependency>
<groupId>com.h2database</groupId>
<artifactId>h2</artifactId>
<version>1.3.170</version>
</dependency>
</dependencies>
[...]
</plugin>
```

Si vous lancez **mvn tomcat7:run** et que vous vous connectez via JMX, vous constatez la présence de votre datasource dans l'annuaire JNDI :



Par contre cela n'empêchera pas quelques logs désagréables au moment de l'invocation de votre service :



Normal, vous n'avez pas créé votre schéma. Pour cela nous allons utiliser le plugin maven-sql ([Lien 30](#)) qui va nous permettre d'exécuter des commandes SQL au démarrage de l'application. Voici la configuration avec des commentaires pour expliquer :

```
<plugin>
<groupId>org.codehaus.mojo</groupId>
<artifactId>sql-maven-
plugin</artifactId>
<version>1.5</version>
<dependencies>
  <!-- Le plugin nécessite lui
aussi de connaître le driver utilisé -->
  <dependency>
<groupId>com.h2database</groupId>
```



```

<artifactId>h2</artifactId>

<version>1.3.170</version>
    </dependency>
</dependencies>
    <!-- La configuration du plugin
doit contenir les informations d'accès à la base
de données -->
    <configuration>

<driver>org.h2.Driver</driver>

<url>jdbc:h2:target/data;MODE=PostgreSQL;DB_CLOSE
_DELAY=-1</url>
    <username>sa</username>
    <password>sa</password>
</configuration>
<executions>
    <!-- on pourrait paramétrer
plusieurs phases, ici on en crée une qui va
exécuter le script de création -->
    <execution>
        <id>create-db</id>
        <phase>process-test-
resources</phase>
        <goals>
            <goal>execute</goal>
</goals>
        <configuration>

<autocommit>true</autocommit>
        <srcFiles>

<srcFile>src/test/resources/schema.sql</srcFile>
        </srcFiles>
    </configuration>
</execution>
</executions>
</plugin>

```

Désormais vous pouvez lancer votre application avec `mvn tomcat7:run` et une datasource JNDI accessible dans votre application.

## 4.2. Jetty

Avec Jetty les étapes sont relativement semblables. Tout d'abord nous créons un fichier qui contient la définition de la datasource : `jetty-ds.xml` ([Lien 31](#)).

```

<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Mort Bay
Consulting//DTD Configure//EN"
"http://jetty.mortbay.org/configure.dtd">
<Configure id="DS"
class="org.eclipse.jetty.webapp.WebAppContext">
    <New
class="org.eclipse.jetty.plus.jndi.Resource">
        <Arg><Ref id="DS"/></Arg>
        <Arg>jdbc/myDb</Arg>
        <Arg>
            <New
class="org.h2.jdbcx.JdbcDataSource">
                <Set
name="uRL">jdbc:h2:target/data;MODE=PostgreSQL;DB
_CLOSE_DELAY=-1;</Set>
                <Set name="user">sa</Set>
                <Set name="password">sa</Set>
            </New>
        </Arg>

```

```

</New>
</Configure>

```

Puis nous faisons référence à cette datasource dans le `pom.xml` :

```

<plugin>
<groupId>org.mortbay.jetty</groupId>
    <artifactId>jetty-maven-
plugin</artifactId>
<version>8.1.5.v20120716</version>
    <configuration>
        <webAppConfig>

<jettyEnvXml>src/test/resources/jetty-
ds.xml</jettyEnvXml>
        </webAppConfig>
    </configuration>
<dependencies>
    <dependency>

<groupId>com.h2database</groupId>
<artifactId>h2</artifactId>
<version>1.3.170</version>
    </dependency>
</dependencies>
</plugin>

```

Le lecteur averti aura remarqué en exécutant `mvn jetty:run` que l'exemple ne fonctionne pas et que l'on obtient l'erreur suivante : **org.h2.jdbc.JdbcSQLException: Database may be already in use: "Locked by another process". Possible solutions: close all other connection(s); use the server mode [90020-170]**

Effectivement après pas mal de tentatives je n'ai pas réussi à faire fonctionner H2 et le plugin Jetty ensemble. Cependant le code source ainsi que l'article sont disponibles sous GitHub donc n'hésitez pas à faire un pull-request si vous parvenez à trouver la solution ;)

## 5. Débogage

Étape inévitable en cours de développement, le débogage avec un conteneur nécessite en général de rajouter des options de lancement dans les propriétés système utilisées au lancement. Les options en questions sont les suivantes :

```

-Xrunjwdp:transport=dt_socket,server=y,suspend=y,
address=8000

```

Et là, bonne nouvelle, pas besoin de modifier les scripts Tomcat avec Maven, il vous suffit de lancer la commande habituelle, mais avec le script `mvnDebug` et non `mvn`. `mvnDebug` est souvent oublié, il s'agit du script de lancement de Maven avec les bonnes propriétés de débogage déjà positionnées.

Récapitulons, si je veux lancer mon appli en debug :

**`mvnDebug tomcat7:run` ou `mvnDebug jetty:run`**

Puis dans Eclipse :

- Debug As ;
- => run configurations ;
- => créer une nouvelle configuration Remote Debug ;

- choisir le projet et le port 8000 ;
- cliquer sur Run.

## 6. Jouez vos tests d'intégration

Vous avez l'habitude de faire des tests unitaires ? C'est très bien, je vous en félicite. Mais testez-vous ensuite vos services sur Tomcat en intégration continue ?

Bien souvent une fois les tests unitaires passés on voit le schéma suivant :

- packaging de l'application au format war ;
- installation manuelle ;
- test manuel par une équipe de QA sur le war déployé.

Et pourquoi ne pas automatiser tout ça et le lancer à chaque build ? (\*)

(\*) En pratique, les tests d'intégration peuvent prendre du temps. Pour éviter de casser votre belle dynamique de test avec des feedbacks rapides vous pouvez aussi splitter TU et TI. Les tests d'intégration joués à part permettent aux TU de renvoyer plus rapidement un résultat.

Pour cela nous allons utiliser le plugin maven-failsafe ([Lien 32](#)) qui va nous permettre de configurer la phase integration-test pour lancer tous les tests de nom : `**/IT*.java`, `**/IT.java`, et `**/ITCase.java`

Vous vous demandez la différence entre maven-surefire et maven-failsafe ? Dites-vous que les deux jouent le même rôle, celui de lancer vos tests. Sauf que maven-failsafe utilise d'autres conventions de nommage pour les tests à lancer et vous facilite ainsi la séparation entre tests unitaires et tests d'intégration.

```
<plugin>
<groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-failsafe-
plugin</artifactId>
  <version>2.13</version>
  <configuration>
  </configuration>
  <executions>
    <execution>
      <id>failsafe-integration-
tests</id>
      <phase>integration-
test</phase>
      <goals>
        <goal>integration-
test</goal>
      </goals>
    </execution>
    <execution>
      <id>failsafe-verify</id>
      <phase>verify</phase>
      <goals>
        <goal>verify</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

Ensuite nous allons configurer les plugin Tomcat et Jetty pour démarrer avant les tests d'intégration.

Pour être capable de conserver les configurations Jetty et Tomcat dans le même POM, j'ai dû utiliser des profils Maven afin de conserver le choix du conteneur utilisé en test d'intégration. L'utilisation de profil n'est pas obligatoire, elle ne le devient que si vous tenez à laisser le choix d'utilisation des deux.

Les deux plugins utilisent la même méthode, il faut se binder(s'attacher) sur les phases d'intégration pour démarrer le conteneur avant les tests et l'éteindre ensuite.

### 6.1. Tomcat

```
<plugin>
<groupId>org.apache.tomcat.maven</groupId>
  <artifactId>tomcat7-
maven-plugin</artifactId>
  <executions>
    <execution>
      <id>run-
tomcat</id>
      <phase>pre-
integration-test</phase>
      <goals>
        <goal>run</goal>
      </goals>
    </execution>
    <execution>
      <id>stop-
tomcat</id>
      <phase>post-
integration-test</phase>
      <goals>
        <goal>shutdown</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

### 6.2. Jetty

```
<plugin>
<groupId>org.mortbay.jetty</groupId>
  <artifactId>jetty-maven-
plugin</artifactId>
  <executions>
    <execution>
      <id>run-jetty</id>
      <phase>pre-
integration-test</phase>
      <goals>
        <goal>run</goal>
      </goals>
    </execution>
    <execution>
      <id>stop-jetty</id>
      <phase>post-
integration-test</phase>
      <goals>
        <goal>stop</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

À noter cependant pour pouvoir appeler le goal stop sur le

plugin Jetty, vous devrez avoir ajouté les clés suivantes dans la configuration du plugin :

```
<stopKey>key</stopKey>
<stopPort>8087</stopPort>
```

Désormais si vous lancez le build avec la commande : **mvn verify -Pjetty** ou **mvn verify -Ptomcat**, vous devriez voir le démarrage de votre serveur avant la phase de tests d'intégration.

Ici avec Jetty :

```
[INFO] << jetty-maven-plugin:8.1.5.v20120716:run (run-jetty) @ developpez-webapp-integration <<
[INFO]
[INFO] --- jetty-maven-plugin:8.1.5.v20120716:run (run-jetty) @ developpez-webapp-integration ---
[INFO] Configuring Jetty for project: developpez-webapp-integration
[INFO] webappSourceDirectory: D:\Developpement\checkout\developpez\maeven\tomcat\4\developpez-webapp-integration\src\main\w
[INFO] Reload Mechanism: automatic
[INFO] classes: D:\Developpement\checkout\developpez\maeven\tomcat\4\developpez-webapp-integration\target\classes
[INFO] Context path: /
[INFO] Tmp directory: D:\Developpement\checkout\developpez\maeven\tomcat\4\developpez-webapp-integration\target\tmp
[INFO] Web services: org.apache.jetsu.webapp.webdefault.wa
[INFO] Web overrides: none
[INFO] Webapp directory: D:\Developpement\checkout\developpez\maeven\tomcat\4\developpez-webapp-integration\src\main\web
[INFO] web.xml file: file://D:/Developpement/checkout/developpez/maeven-tomcat/4/developpez-webapp-integration/src/main/web
2013-02-24 19:27:19.985 INFO org.eclipse.jetty: Server: Jetty 8.1.5.v20120716
2013-02-24 19:27:19.988 INFO org.eclipse.jetty: Configuration No Transaction manager found - if your webapp requires one, please c
2013-02-24 19:27:21.895 INFO /No Spring WebApplicationInitializer types detected on classpath
2013-02-24 19:27:22.015 INFO org.springframework.context: Refreshing Root WebApplicationContext. Startup date [Sun F
2013-02-24 19:27:22.015 INFO org.springframework.context: Loading XML bean definitions from ServletContext resource
2013-02-24 19:27:22.016 INFO org.springframework.context: Loading XML bean definitions from class path resource [de
2013-02-24 19:27:22.160 INFO /Initializing Spring root WebApplicationContext
19:27:22.162 [main] INFO o.s.web.context.ContextLoader - Root WebApplicationContext: initialization started
19:27:22.216 [main] INFO o.s.w.s.c.WebApplicationContext - Refreshing Root WebApplicationContext. Startup date [Sun F
19:27:22.258 [main] INFO o.a.b.f.xal.XmlBeanDefinitionReader - Loading XML bean definitions from ServletContext resource
19:27:22.398 [main] INFO o.a.b.f.xal.XmlBeanDefinitionReader - Loading XML bean definitions from class path resource [de
19:27:22.688 [main] INFO o.a.b.f.a.AutowiredAnnotationBeanPostProcessor - JSR-330 'java:inject' inject annotation found
19:27:22.692 [main] INFO o.a.b.f.a.AutowiredAnnotationBeanPostProcessor - Preinstantiating singletons in org.springframework.beans
org.springframework.beans.factory.config.PropertyPlaceholderConfigurer
19:27:22.948 [main] INFO org.apache.cxf.endpoint.ServerImpl - Setting the server's publish address to be /
19:27:22.899 [main] INFO o.s.web.context.ContextLoader - Root WebApplicationContext: initialization completed in 834 ms
2013-02-24 19:27:22.899 INFO org.springframework.context: Loading XML bean definitions from class path resource [de
2013-02-24 19:27:23.072 INFO org.springframework.connector: Started SelectChannelConnector0.0.0.0:8088
[INFO] Started Jetty Server
[INFO]
[INFO] --- maven-failsafe-plugin:2.13:integration-test (failsafe-integration-tests) @ developpez-webapp-integration ---
[INFO] File encoding has not been set, using platform encoding cp1252, i.e. build is platform dependent!
[INFO]
[INFO] --- jetty-maven-plugin:8.1.5.v20120716:stop (stop-jetty) @ developpez-webapp-integration ---
```

À vous de jouer maintenant pour coder un test d'intégration qui profite de cette astuce.

## 7. Précompilation des JSP

Mettons cette fois que votre objectif ne soit plus uniquement le développement, mais le lancement de votre application en production. Voici quelques astuces intéressantes.

Vous l'aurez sans doute remarqué, la première fois que vous arrivez sur une page après avoir relancé votre Tomcat celle-ci est plus lente à s'afficher. C'est parce que Tomcat compile vos JSP en servlets lors de leur première visite. En plus d'être relativement désagréable pour vos utilisateurs, c'est aussi un peu tardif pour découvrir des erreurs de compilations, vous ne trouvez pas ?

Application du principe fail fast ([Lien 33](#)), on va mettre en place la précompilation des JSP.

### 7.1. Tomcat

Première étape, déclarez le plugin de compilation de JSP :

```
<plugin>
<groupId>org.codehaus.mojo</groupId>
<artifactId>jspc-maven-
plugin</artifactId>
<executions>
<execution>
<id>jspc</id>
<goals>
<goal>compile</goal>
</goals>
</execution>
</executions>
</plugin>
```

Si vous compilez votre application, vous remarquerez que désormais les JSP sont compilées pendant la phase de compilation (logique c'est vrai, mais ce n'était pas le cas sans ce plugin) :

```
[INFO] --- maven-compiler-plugin:2.5:compile (default-compile) @ developpez-webapp-jsp ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- jspc-maven-plugin:1.4:compile (jspc) @ developpez-webapp-jsp ---
2013-02-24 19:27:24.721 [main] DEBUG org.apache.jasper.compiler.Compiler - Generated D:/Developpement/checkout/developpez/maeven\tomcat
2013-02-24 19:27:24.721 [main] DEBUG org.apache.jasper.compiler.Compiler - Compiled D:/Developpement/checkout/developpez/maeven\tomcat
2013-02-24 19:27:24.721 [main] DEBUG org.apache.jasper.compiler.SnoopInstaller - constant pool count: 107
2013-02-24 19:27:24.721 [main] DEBUG org.apache.jasper.compiler.SnoopInstaller - 1 read class attr -- jsp/index.jsp
```

Cette étape a pour but de précompiler chaque JSP en une classe de Servlet. Les résultats de cette opération sont :

- l'ensemble de vos servlets compilés dans le répertoire target/classes ;
- un fichier jspweb.xml qui dérive de votre fichier web.xml initial et qui contient en plus la déclaration de chaque servlet.

Il faut donc modifier la configuration du plugin war pour qu'il utilise le fichier jspweb.xml construit par le plugin de compilation des JSP.

```
<plugin>
<groupId>org.apache.maven.plugins</groupId>
<artifactId>maven-
war-plugin</artifactId>
<version>2.0</version>
<configuration>
<webXml>${
basedir}/target/jspweb.xml</webXml>
</configuration>
</plugin>
```

Je vous laisse observer le contenu du war afin de vérifier qu'il contient bien la servlet déjà compilée et le fichier web.xml avec la déclaration attendue.

### 7.2. Jetty

Le principe pour Jetty est identique, vous pouvez donc vous reporter au paragraphe précédent pour comprendre le fonctionnement.

Spécifiquement, voici la configuration du plugin de compilation :

```
<plugin>
<groupId>org.mortbay.jetty</groupId>
<artifactId>jetty-jspc-maven-
plugin</artifactId>
<version>8.1.5.v20120716</version>
<executions>
<execution>
<id>jspc</id>
<goals>
<goal>jspc</goal>
</goals>
</execution>
</executions>
</plugin>
```

Et le plugin war modifié (attention, le plugin Jetty crée un fichier web.xml et le plugin Tomcat crée un fichier jspweb.xml !!)

```
<plugin>
<groupId>org.apache.maven.plugins</groupId>
<artifactId>maven-war-
plugin</artifactId>
<version>2.0</version>
<configuration>
```

```
<webXml>$
{basedir}/target/web.xml</webXml>
</configuration>
</plugin>
```

Astuce : cette étape étant coûteuse en temps, vous pouvez la déplacer dans un profil pour ne pas la jouer en phase de développement.

## 8. Ressources

- Code de l'article sous GitHub : [Lien 34](#)
- L'article sous GitHub : [Lien 35](#)
- Le très bon blog de Khan (jetoile) qui parle aussi de Maven : [Lien 36](#)
- Un article sur le débogage avec Eclipse et une appli Java sur developpez.com : [Lien 37](#)

## 9. Conclusion

Voilà, nous avons fait le tour de quelques plugins Maven nécessaires pour améliorer notre productivité.

- Nous avons mis en œuvre le « checkout and run » afin que chaque nouvel entrant sur le projet puisse rapidement démarrer.
- Nous avons centralisé des configurations dans notre pom.xml pour éviter à chacun de refaire son intégration et pour éviter aussi le syndrome « homme-clé », le type seul capable de monter votre environnement de travail.
- Nous savons comment déboguer, jouer nos tests.
- Et nous avons même vu une astuce bien pratique pour nos performances en production avec la précompilation des JSP.

Cet article sera suivi par d'autres dans la même veine expliquant notamment comment faire du SOA ou du développement JavaScript avec Maven. *Stay tuned.*

Retrouvez l'article de Hugo Lassiège en ligne : [Lien 38.](#)





### IntelliJ IDEA sort en version 12.1 l'environnement de développement « Java intelligent » introduit le support de JavaFX 2

IntelliJ IDEA, l'environnement de « développement polyglotte basé sur la JVM » maintenu par JetBrains, a atteint sa version 12.1.

Cette mise à jour majeure se distingue essentiellement par la prise en charge de JavaFX 2. Le support de la plateforme de développement d'applications riches d'Oracle permet de mettre à la disposition des développeurs des outils pour FXML markup, custom CSS, la complétion de code, la navigation et la recherche, le refactoring et l'intégration avec SceneBuilder.

Présentation du support de JavaFX 2 : [Lien 39](#)

Darcula, la nouvelle interface sobre et élégante introduite par la version 12.0, évolue pour offrir une meilleure expérience aux développeurs. Le support de Retina a été amélioré. L'EDI intègre déjà le support de Java 8, des améliorations pour le concepteur d'IU Android, un nouveau compilateur double-performance et de nouveaux outils de bases de données.

À ces nouveautés majeures s'ajoutent d'autres améliorations intéressantes comme :

- le mode plein écran pour Windows ;
- le support des frameworkq Spring 3.2 et Play 2.1 ;
- la prise en charge de Groovy 2.1 ;
- des améliorations pour le support de Scala ;
- la prise en charge du SDK Adobe Gaming ;
- le débogueur pour CoffeeScript et TypeScript via les cartes de références.

La mise à jour IntelliJ IDEA 12.1 est disponible gratuitement pour les possesseurs d'une licence IntelliJ IDEA 12 Ultimate.

*Commentez la news de Hinault Romaric en ligne : [Lien 40](#)*

### Java 8 sortira le 18 mars 2014, Oracle publie une nouvelle roadmap pour le langage

Oracle vient de définir officiellement un nouveau calendrier pour Java SE 8, dont la date de sortie initiale a été retardée pour corriger les failles dans la plateforme (voir section « Retrouvez le dossier complet de la rédaction »).

Java fait depuis plusieurs mois face à une vague de failles de sécurité exploitables via les navigateurs. La priorité d'Oracle est de ramener la confiance des utilisateurs en accordant une priorité absolue à la correction des vulnérabilités.

Ainsi, les développeurs n'ont pas pu se concentrer sur l'intégration des nouvelles fonctionnalités à Java 8, repoussant ainsi sa sortie au 18 mars 2014, selon un récent billet de blog de Mark Reinhold, le responsable du projet chez Oracle.

La nouvelle roadmap du langage prévoit le gel des fonctionnalités de JDK 8 le 23 mai 2013, la sortie d'une préversion pour les développeurs en début septembre et la publication de la Release Candidate en fin janvier 2014.

Pour rappel, Java 8 introduira comme nouveautés : les expressions lambda, le moteur JavaScript Nashorn, les annotations, la nouvelle API « date and time », la convergence des JVM et bien plus. Le projet Jigsaw pour la mise en place d'un système de module a été repoussé à Java 9.

*Commentez la news de Hinault Romaric en ligne : [Lien 41](#)*

### Tutoriel sur l'utilisation de l'API Java RMI avec le protocole IIOP

RMI (Remote Method Invocation) est une API Java basée sur les ORB (Object Request Broker) qui permet l'invocation d'objets distants entre processus. Nous verrons comment utiliser RMI pour le rendre compatible avec le protocole IIOP et permettre ainsi des appels entre applications basées sur l'architecture CORBA et développées avec d'autres langages que Java.

Dans un premier temps, nous ferons une petite présentation de RMI. Puis nous verrons comment mettre en pratique RMI au niveau développement et mise en exploitation.

#### 1. Présentation

L'API RMI (Remote Method Invocation) permet d'exécuter des instructions à travers le réseau. Elle met en œuvre tous les mécanismes nécessaires pour que deux applications puissent communiquer entre elles au travers d'objets Java. Ainsi, il sera possible d'appeler un objet instancié dans une autre application et pouvant même se trouver sur une autre machine, ceci de manière totalement transparente.

RMI est l'une des solutions les plus simples pour communiquer à travers un réseau. Néanmoins, cette solution peut s'avérer compliquée à partir du moment où il faut traverser des pare-feux. Dans ce cas, et bien qu'il existe des solutions pour contourner ce problème, la solution des Web Services est mieux adaptée.

Par défaut, RMI utilise le protocole JRMP (Java Remote Method Protocol) qui ne marche qu'entre des applications écrites en Java. Mais si l'on veut communiquer entre des applications écrites dans divers langages, il faut utiliser le protocole IIOP (Internet Inter-ORB Protocol) qui est le protocole utilisé par CORBA (Common Object Request Broker Architecture). Il faut bien garder à l'esprit que l'utilisation du protocole JRMP est beaucoup plus simple que le protocole IIOP.

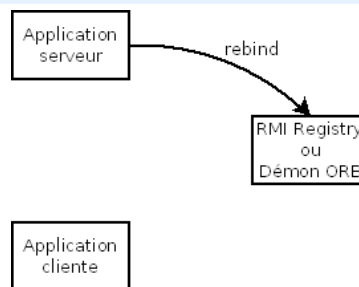
Au niveau du développeur, RMI rend complètement transparent toute la problématique des échanges entre processus. Ça a son avantage, mais aussi son inconvénient surtout lorsqu'on rencontre des problèmes réseau.

En principe, nous allons avoir trois acteurs :

- le RMI Registry (pour JRMP) ou le démon ORB (pour IIOP) ;
- une application jouant le rôle de serveur ;
- une application jouant le rôle de client.

Le RMI Registry ou le démon ORB sont des applications (livrées avec Java) permettant de fournir un service réseau (sur le port TCP 1099 pour le RMI Registry) chargé d'aiguiller chaque requête RMI vers le bon objet distant.

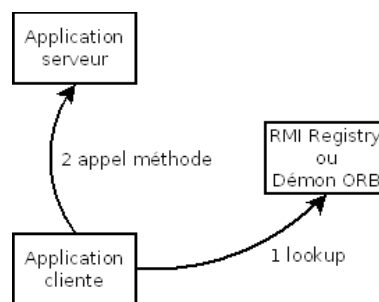
La première étape, consiste pour l'application serveur de publier (rebind) un objet distant auprès du RMI Registry ou du démon ORB :



Avant la publication d'un objet (rebind), l'application serveur doit rendre l'objet distant (exportObject). C'est lors de cette opération qu'un numéro de port TCP va être assigné à l'objet. Chaque objet distant aura son propre numéro de port. Par défaut, il est choisi aléatoirement rendant ainsi le passage à travers les pare-feux impossible. Mais avec JRMP, il est possible de fixer ce numéro de port.

Il est recommandé d'enregistrer auprès du RMI Registry ou du démon ORB (rebind) uniquement des fabriques (factory). Ainsi on aurait qu'une seule fabrique par application qui donnerait accès à tous les autres objets distants.

L'étape suivante consiste pour l'application cliente à appeler une méthode de l'objet distant :



Pour cela, l'application cliente interroge le RMI Registry ou le démon ORB pour obtenir le service à appeler (lookup). Puis, elle appelle la méthode de l'objet distant situé sur l'application serveur au numéro de port correspondant.

Rien n'empêche d'avoir deux applications qui communiquent entre elles en jouant chacune le rôle de serveur et de client. Ce type d'architecture est tout de même à proscrire.

Pour éviter d'avoir des applications qui soient en même temps client et serveur, il est possible d'utiliser le pattern

Observateur. Le client envoie au serveur un observateur, préalablement rendu distant (exportObject). Pour envoyer des informations au client, le serveur utilisera l'observateur. Cette solution est beaucoup plus propre, même si au niveau des flux réseau ça ne change pas grand-chose.

Dans le cas où sur une machine il n'y a qu'une seule application serveur qui utilise le RMI Registry, il serait peut-être intéressant de l'intégrer dans l'application serveur.

Pour intégrer RMI dans des applications, il faut :

- voir la segmentation en plusieurs fichiers JAR ;
- modifier le fichier POM de Maven pour intégrer les spécificités liées à RMI, et plus particulièrement au protocole IIOP ;
- si nécessaire, ajouter du code pour intégrer le RMI Registry dans l'application serveur.

L'implémentation d'un objet distant suit le mode opératoire suivant :

- création de l'interface de l'objet ;
- création de la classe de l'objet avec les traitements ;
- ajout du code pour rendre l'objet distant (exportObject) ;
- si l'objet n'est pas obtenu à partir d'une fabrique distante, ajout du code pour le publier auprès du RMI Registry ou du démon ORB (rebind).

Pour appeler une méthode d'un objet distant, il faut :

- ajouter le code pour obtenir l'instance de l'objet (lookup) ;
- ajouter le code pour appeler la méthode.

## 2. Mise en pratique

Nous allons prendre comme exemple, un objet chargé de lire un fichier sur la machine où est hébergée l'application serveur. Dans un premier temps nous aborderons RMI avec le protocole JRMP. Puis nous verrons comment basculer vers le protocole IIOP.

- Nous allons d'abord créer notre objet distant ().
- Ensuite, nous verrons comment, avec le protocole JRMP, publier notre objet distant auprès du RMI Registry ().
- Nous parlerons ensuite du RMI Registry avec notamment les explications pour l'intégrer dans l'application serveur ().
- Ensuite, nous verrons comment, avec le protocole JRMP, obtenir une instance de notre objet distant et exécuter une de ses méthodes ().
- Nous verrons ensuite ce qu'il faut faire pour basculer du protocole JRMP vers le protocole IIOP ().
- Ensuite, nous verrons comment segmenter notre application serveur et client en plusieurs fichiers JAR ().
- Nous aborderons par la suite, et ceci uniquement pour le protocole IIOP, les spécificités à apporter au fichier POM de Maven pour prendre en compte la génération des stubs, skeletons et IDL ().

- Et enfin, nous verrons comment mettre en production notre exemple ().

## 3. Création d'objets distants

La mise en œuvre de RMI commence par la création d'objets distants au sein d'une application serveur. Ces objets auront :

- une partie spécification sous forme d'interface héritant de java.rmi.Remote ([Lien 42](#)) ;
- une partie implémentation.

Par exemple :

### FileInterface.java

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface FileInterface extends Remote {

    byte[] downloadFile (String fileName) throws
    RemoteException;

}
```

### FileImpl.java

```
import java.io.*;
import java.rmi.*;

public class FileImpl implements FileInterface,
Serializable {

    private static final long serialVersionUID =
-115071828855576992L;

    private String name;

    public FileImpl (String s) throws
RemoteException {
        super();
        name = s;
    }

    public byte[] downloadFile (String fileName) {
        byte[] buffer;
        try {
            File file = new File(fileName);
            buffer = new byte[(int)file.length()];
            BufferedInputStream input = new
BufferedInputStream(new
FileInputStream(fileName));
            input.read(buffer,0,buffer.length);
            input.close();
        }
        catch(Exception e) {
            System.out.println("FileImpl:
"+e.getMessage());
            e.printStackTrace();
            buffer = null;
        }
        return(buffer);
    }
}
```

Pour éviter des erreurs de type **MarshalException**, il ne faudra surtout pas oublier de rendre Serializable la classe de l'objet distant et d'y indiquer un serialVersionUID.

#### 4. Publication d'un objet avec JRMP

Une fois les objets créés, il ne reste plus qu'à les publier auprès d'un service appelé RMI Registry. Pour cela, on aura besoin des classes `java.rmi.server.UnicastRemoteObject` ([Lien 43](#)) et `java.rmi.Naming` ([Lien 44](#)) :

```
final Remote obj = new FileImpl("myFile.txt");
final Remote remoteStub =
UnicastRemoteObject.exportObject (obj);
Naming.rebind ("//localhost/FileServer",
remoteStub);
```

La méthode `exportObject` permet de rendre distant un objet et la méthode `rebind` permet d'enregistrer l'objet auprès du RMI Registry.

Ici, un numéro de port TCP sera assigné aléatoirement à notre objet. Chaque objet aura son numéro de port. L'assignation aléatoire de ce numéro peut devenir problématique si l'on veut traverser des pare-feux. Dans ce cas, il sera préférable de fixer le numéro de port (par exemple 19001) :

```
final Remote remoteStub =
UnicastRemoteObject.exportObject (obj, 19001);
```

La communication RMI entre processus est toujours problématique lorsqu'il s'agit de traverser des pare-feux. Pensez à fixer un numéro de port TCP pour chacun de vos objets distants. Choisissez des numéros qui se suivent pour faciliter la configuration des pare-feux. N'oubliez pas aussi d'ouvrir un port TCP pour le RMI Registry (1099 par défaut).

Si le nom de l'objet (« FileServer » dans notre exemple), passé dans la méthode `rebind`, est déjà utilisé, une erreur de type **AlreadyBoundException** se produit.

Pour ne plus publier un objet distant :

```
Naming.unbind ("//localhost/FileServer");
```

Pensez à envoyer des traces dans vos logs à chaque fois que des objets sont publiés auprès du RMI Registry. Ce dernier étant avare d'information, vos logs seront très précieux en cas de problème lors de la mise en production.

#### 5. Le RMI Registry

Avant de lancer notre application serveur (ou toute application publiant des objets distants via RMI), il faudra s'assurer que le RMI Registry soit bien lancé :

```
$JAVA_HOME/bin/rmiregistry
```

Le RMI Registry ne redonne pas la main une fois lancé. Pour l'arrêter, il suffit d'interrompre le processus par l'appui simultané des touches Ctrl + C. Il est possible de lancer le RMI Registry sur un autre numéro de port :

```
$JAVA_HOME/bin/rmiregistry 18001
```

Dans ce cas précis, il faudra indiquer le bon numéro de port dans les URL lors de chaque connexion au RMI

Registry :

```
Naming.rebind ("//localhost:18001/FileServer",
remoteStub);
```

Dans un système avec une seule application serveur, il serait peut-être intéressant d'embarquer le RMI Registry dans notre application. Pour cela, il suffit de créer une instance de `java.rmi.registry.Registry` ([Lien 45](#)) à l'aide de la classe `java.rmi.registry.LocateRegistry` ([Lien 46](#)) :

```
final Registry registry =
LocateRegistry.createRegistry (18001);
```

Ici, notre RMI Registry embarqué dans notre application écoute sur le port TCP 18001.

Pour arrêter le RMI Registry :

```
UnicastRemoteObject.unexportObject (registry,
true);
```

#### 6. Accès à un objet distant avec JRMP

Une application cliente pourra obtenir une instance de notre objet distant :

```
final FileInterface obj = (FileInterface)
Naming.lookup ("//localhost/FileServer");
```

et exécuter des méthodes de cet objet :

```
final byte[] filedata = obj.downloadFile
(fileName);
```

Pour que cela fonctionne, notre application cliente devra avoir dans son CLASSPATH un JAR contenant l'interface `FileInterface` **sans son implémentation** (classe `FileImpl`).

Comme pour la publication d'objet, il est intéressant d'envoyer des traces dans vos logs à chaque fois que des instances sont demandées auprès du RMI Registry. Vos logs seront très utiles en cas de problème lors de la mise en production.

#### 7. Compatibilité IIOP

Pour basculer vers le protocole IIOP :

- au lieu d'utiliser un `UnicastRemoteObject` ([Lien 47](#)), on utilisera un `PortableRemoteObject` ([Lien 48](#)) ;
- au lieu d'utiliser un `java.rmi.Naming` ([Lien 49](#)), on utilisera un `javax.naming.InitialContext` ([Lien 50](#)).

Ainsi, la publication d'un objet distant ressemblera à ceci :

```
final Properties properties = new Properties ();
properties.setProperty
("java.naming.factory.initial",
"com.sun.jndi.cosnaming.CNCTxFactory");
properties.setProperty
("java.naming.provider.url",
"iiop://localhost:1050");
final Context namingContext = new InitialContext
(properties);
final Remote obj = new FileImpl("myFile.txt");
PortableRemoteObject.exportObject (obj);
namingContext.rebind
```



```
("org.minetti.test,type=FileServer", obj);
```

Avec IIOP, il n'est pas possible de fixer les numéros de port TCP pour chaque objet distant.

Pour ne plus publier un objet distant :

```
namingContext.unbind  
("org.minetti.test,type=FileServer");  
PortableRemoteObject.unexportObject (obj);
```

Pour qu'une application cliente puisse obtenir une instance de notre objet distant :

```
final Properties properties = new Properties ();  
properties.setProperty  
("java.naming.factory.initial",  
"com.sun.jndi.cosnaming.CNCtxFactory");  
properties.setProperty  
("java.naming.provider.url",  
"iiop://localhost:1050");  
final Context namingContext = new InitialContext  
(properties);  
final Object objref = namingContext.lookup  
("org.minetti.test,type=FileServer");  
final FileInterface obj = (FileInterface)  
PortableRemoteObject.narrow (objref,  
FileInterface.class);
```

Avec IIOP, il est impératif de générer le squelette et le stub :

```
$JAVA_HOME/bin/rmic -iiop -verbose FileImpl
```

Dans notre exemple, cette commande va générer les classes suivantes :

- **FileInterface\_Stub.class** pour le stub à insérer dans le JAR de l'application cliente et serveur ;
- **FileImpl\_Tie.class** pour le squelette à insérer dans le JAR de l'application serveur.

Si l'on veut développer des applications clientes dans d'autres langages, il faut générer les IDL de toutes les classes utiles pour nos communications distantes :

```
$JAVA_HOME/bin/rmic -idl FileImpl
```

Au lieu d'utiliser le RMI Registry, on utilisera un démon ORB comme par exemple orbd, fourni avec Java :

```
$JAVA_HOME/bin/orbd -ORBInitialPort 1050  
-ORBInitialHost localhost
```

## 8. Choix de l'architecture logicielle

D'une manière générale, il est intéressant de segmenter notre application cliente et serveur en trois fichiers JAR :

- un fichier JAR commun qui se trouvera référencé dans le CLASSPATH de l'application cliente et serveur ;
- un fichier JAR (ou WAR dans le cas d'une application web) pour l'application serveur ;
- un fichier JAR (ou WAR dans le cas d'une application web) pour l'application cliente.

On mettra dans le fichier JAR commun :

- les interfaces de tous les objets distants.

Dans le fichier JAR (ou WAR) de l'application serveur, on mettra :

- les classes des objets distants (implémentation) ;
- les stubs des objets distants (uniquement avec IIOP) ;
- les squelettes des objets distants (uniquement avec IIOP).

Et enfin, on mettra dans le fichier JAR (ou WAR) de l'application cliente :

- les stubs des objets distants (uniquement avec IIOP).

On aurait pu mettre les stubs dans le fichier JAR commun. On verra par la suite (dans le ) qu'il est plus simple de générer les stubs dans les JAR des applications.

Au niveau de l'application cliente, il faudra éviter de mettre dans le CLASSPATH le JAR de l'application serveur. Le client ne doit pas avoir accès à l'implémentation des objets distants. Cela nous évitera d'avoir des accès directs à nos objets.

## 9. Industrialisation avec Maven

Maven est un outil permettant d'industrialiser la création des différents JAR. Nous n'aborderons pas ici toute la configuration de Maven mais uniquement les spécificités liées à RMI et plus particulièrement au protocole IIOP.

Pour en savoir plus sur Maven :

- Introduction à Maven 2 : [Lien 51](#) ;
- Utiliser Maven 2 : [Lien 52](#).

Pour le protocole JRMP, il n'y a aucune difficulté à paramétrer Maven puisqu'il n'y a pas de commande particulière à lancer lors de la fabrication des JAR.

Pour le protocole IIOP, c'est une autre histoire puisqu'il faut générer les stubs, les squelettes et les IDL. Au niveau du projet de l'application serveur, on ajoutera ceci dans le fichier pom.xml :

```
pom.xml  
...  
<!-- RMIC executor -->  
<plugin>  
  <groupId>org.codehaus.mojo</groupId>  
  <artifactId>rmic-maven-plugin</artifactId>  
  <version>1.1</version>  
  <executions>  
    <execution>  
      <id>rmic-process-classes</id>  
      <goals>  
        <goal>rmic</goal>  
      </goals>  
      <configuration>  
        <iiop>true</iiop>  
        <includes>  
<include>org/minetti/test/rmi/FileImpl.class</inc  
lude>  
        </includes>  
        <outputDirectory>${  
{project.build.outputDirectory}</outputDirectory>  
        </configuration>  
      </execution>
```

```

    <execution>
      <id>rmic-process-idls</id>
      <goals>
        <goal>rmic</goal>
      </goals>
      <configuration>
        <idl>true</idl>
        <includes>
<include>org/minetti/test/rmi/FileImpl.class</inc
lude>
          </includes>
          <outputDirectory>${
project.build.directory}/idls</outputDirectory>
        </configuration>
      </execution>
    </executions>
  </plugin>

  <!-- Application & IDLs assembling -->
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-assembly-
plugin</artifactId>
    <version>2.3</version>
    <executions>
      <execution>
        <id>make-idls-assembly</id>
        <phase>install</phase>
        <goals>
          <goal>single</goal>
        </goals>
        <configuration>
          <descriptors>
<descriptor>idl.xml</descriptor>
          </descriptors>
          <finalName>TestServer-idl-$(
project.version)</finalName>
          </configuration>
        </execution>
      </executions>
    </configuration>

    <appendAssemblyId>false</appendAssemblyId>
    <outputDirectory>${
project.build.directory}</outputDirectory>
    </configuration>
  </plugin>
  ...

```

#### idl.xml

```

<assembly
xmlns="http://maven.apache.org/plugins/maven-
assembly-plugin/assembly/1.1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
xsi:schemaLocation="http://maven.apache.org/plugi

```

```

ns/maven-assembly-plugin/assembly/1.1.0
http://maven.apache.org/xsd/assembly-1.1.0.xsd">
  <id>idl</id>

  <formats>
    <format>tar.gz</format>
    <format>zip</format>
  </formats>

  <fileSets>
    <fileSet>
      <directory>${
project.build.directory}/idls</directory>
      <outputDirectory></outputDirectory>
    </fileSet>
  </fileSets>
</assembly>

```

L'exécution « rmic-process-classes » permet de créer le stub et le skeleton de la classe org.minetti.test.rmi.FileImpl.

L'exécution « rmic-process-idls » permet de créer tous les IDL et les range dans le répertoire « idls ».

L'exécution « stub-package » permet de créer un fichier JAR avec le classifieur « stub » et contenant tous les stubs générés par « rmic-process-classes ».

Le dernier plugin (« maven-assembly-plugin ») permet de générer un fichier zip et tar.gz contenant tous les IDL générés par « rmic-process-idls ».

Au niveau du projet de l'application cliente, on ajoutera ceci dans le fichier pom.xml :

#### pom.xml

```

...
<!-- Stub classes importation & Packages
installation -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-dependency-
plugin</artifactId>
  <executions>
    <execution>
      <goals>
        <goal>unpack</goal>
      </goals>
    </configuration>
    <artifactItems>
      <artifactItem>
<groupId>org.minetti</groupId>
<artifactId>TestServer</artifactId>
        <version>$(
project.version)</version>
        <type>jar</type>
<classifier>stub</classifier>
        <outputDirectory>$(
project.build.outputDirectory)</outputDirectory>
      </artifactItem>
    </artifactItems>
    </configuration>
  </executions>
</plugin>

```

...

Ce plugin va nous permettre de récupérer le fichier JAR avec le classifieur « stub » et contenant tous les stubs pour les intégrer dans le JAR de l'application cliente. Pour que ça marche, il faudra bien sûr que le JAR de l'application serveur soit généré avant celui de l'application cliente.

## 10. Mise en exploitation

Avant de lancer les différentes applications, il est impératif de lancer le RMI Registry s'il n'est pas embarqué dans une des applications (voir ) ou le démon ORB de votre choix si vos applications communiquent avec le protocole IIOP :

### JRMP

```
rmiregistry
```

### IIOP

```
orbd -ORBInitialPort 1050 -ORBInitialHost localhost
```

Le RMI Registry ou le démon ORB doit se trouver obligatoirement sur la même machine que la ou les applications serveur.

Si les numéros de port TCP ont été fixés pour chaque objet distant, le passage au travers des pare-feux ne posera pas de problème. Autrement, il sera très difficile de traverser le moindre pare-feu, car les numéros de port TCP seront alloués aléatoirement.

Ce problème peut être contourné par l'utilisation de « GIOP proxy » avec le protocole IIOP ou de « RMI proxy » avec le protocole JRMP.

Ensuite, il faudra lancer la ou les applications publiant des objets distants.

En cas de problème, le RMI Registry étant peu bavard, il

faudra se référer aux logs des applications serveur et client. D'où la nécessité d'accorder une attention toute particulière aux traces envoyées dans les logs lors de la publication d'objets distants et de leurs accès.

Si le RMI Registry n'a pas été démarré ou qu'il est impossible de l'atteindre, une erreur de type **NotBoundException** se produit lors de la publication des objets distants. Pour le démon ORB, on aura une erreur de type **CommunicationException**.

Et enfin, on lancera en dernier toutes les applications accédant aux objets distants.

Après une mise en production, une erreur de type **MarshalException** peut se produire lors d'une tentative d'accès à un objet distant. Cela peut provenir d'un oubli d'une sérialisation d'une classe mais aussi d'une mauvaise construction des JAR. L'interface RMI a dû changer et l'une des applications n'a pas été mise à jour.

## 11. Conclusion

La mise en œuvre de RMI est relativement simple. Les choses se compliquent un peu lorsqu'il s'agit de rendre nos communications entre processus compatibles avec le protocole IIOP. Dans ce cas, il faut générer le skeleton, le stub, et éventuellement les classes au format IDL. Mais la compatibilité IIOP permet la communication entre applications construites dans des langages autres que Java. Avant de commencer le développement, il faudra réfléchir sur l'architecture réseau. Bien se poser la question si les flux réseau doivent traverser des pare-feux.

Aussi, il faudra se poser la question si des applications seront écrites dans d'autres langages.

Retrouvez l'article de Jean-Philippe Minetti en ligne : [Lien 53](#)



### Google I/O : « Nous sommes à 1 % de ce qui est possible », le PDG de Google vante les innovations de la société

Le Google I/O, le rendez-vous annuel de Google destiné aux développeurs, bat son plein. Depuis mercredi dernier, Google dévoile ses récentes innovations.

La keynote d'ouverture a été présentée par Larry Page, PDG de Google, qui, malgré une paralysie partielle des cordes vocales, est monté d'un cran pour dévoiler la vision de l'évolution IT de son entreprise.

« Aujourd'hui, nous allons encore un peu plus gratter la surface », a déclaré Page. « Google travaille sur tellement d'innovations. J'ai eu la chair de poule à ce sujet ».



La première et plus remarquable à l'ouverture de l'événement a été la présentation d'Android Studio ([Lien 54](#)), un nouvel environnement de développement pour Android conçu par la firme, permettant de visualiser les UI des applications sur diverses tailles d'appareils.

Ensuite s'en est suivi la présentation d'une pléthore de nouvelles API (Activity Recognition, Google+ Sign-in, Maps (v2) et Locations, etc. ([Lien 55](#))) et des évolutions de plusieurs services, notamment une refonte complète de Google+ ([Lien 56](#)), des améliorations pour Google Play Game Services ou encore Google Cloud Messaging.

De nouveaux services ont également vu le jour, dont le service de musique « Google Play Music All Access » : [Lien 57](#). La plateforme Cloud App Engine ([Lien 58](#)) s'est ouverte au PHP, le nouveau Google Maps pour PC a été dévoilé ([Lien 59](#)), ainsi que des préversions pour Android et iOS.

La recherche, le cœur de métier de la société, n'était pas en reste, avec des évolutions de la recherche sémantique, la recherche vocale et l'ouverture de Google Now aux PC : [Lien 60](#).

« Chaque fois que nous faisons quelque chose de fou, nous avons fait des progrès », se félicite Page. « Aujourd'hui, nous sommes devenus une entreprise audacieuse. Nous sommes à 1 % de ce qui est possible ».

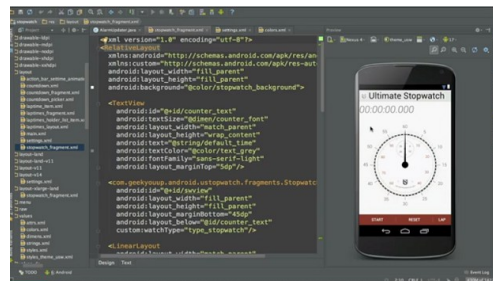
Un exemple de projet fou est, par exemple, les lunettes de réalité augmentée, les Google Glass ([Lien 61](#)), dont les premiers prototypes pour les développeurs sont disponibles, et ceux-ci peuvent déjà créer des applications pour le dispositif.

Cependant, cette course vers l'innovation passe par l'abandon de plusieurs projets pas assez rentables pour la société. Les développeurs ont encore un souvenir amer de la fin de Google Reader et de plusieurs autres services.

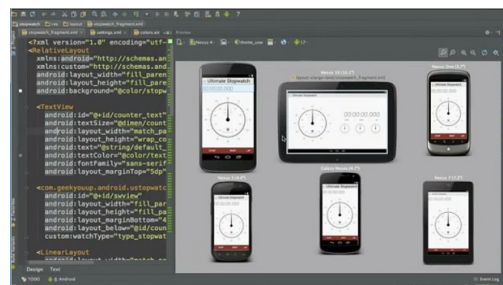
Commentez la news de Hinault Romaric en ligne : [Lien 62](#)

### Google annonce un nouvel EDI pour Android : Android Studio Au Google I/O, il s'appuiera sur IntelliJ

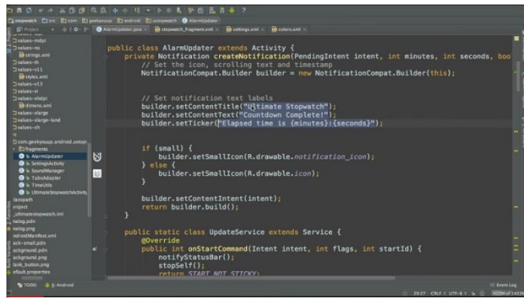
C'est la grande annonce qui a suscité le plus d'enthousiasme au Google I/O, la grand-messe de Google dédiée aux développeurs. L'OS mobile va avoir un nouvel EDI, qui s'appuie sur IntelliJ de JetBrains : Android Studio.



L'outil offre la possibilité de prévisualiser les interfaces des applications sur diverses tailles d'appareils. Et dans différentes langues.



Google n'en a pas dit beaucoup plus (histoire d'attiser la curiosité ?) mais une démonstration a permis de voir qu'il permettait des éditions en temps réel.



« Cet IDE a été conçu pour vous permettre, développeurs d'applications, de travailler plus rapidement et de manière plus productive », explique Google.

Commentez la news de Gordon Fowler en ligne : [Lien 63](#)

## Google Glass : Winky permet la prise de photo par un clignement de l'œil son code source disponible, les lunettes high-tech pourraient être synchronisées avec l'iPhone

Mike DiGiovanni, un développeur chez Roundarch Isobar, a publié le code source de Winky sur GitHub. Winky est l'application permettant aux Google Glass de prendre des photos ou des vidéos d'un clignement de l'œil.



« Winky change réellement les choses. Vous pourrez peut-être penser qu'il n'est pas difficile de dire "OK, Glass prenez une photo" ou même simplement appuyer sur un bouton. Mais c'est un changement de contexte qui vous transporte hors du moment. J'ai pris plus de photos aujourd'hui que je ne l'ai fait ces cinq derniers jours grâce à cette application » dit-il sur sa page Google+.

Une vidéo est même disponible sur sa page montrant Winky en action ; l'application ferait la différence entre un clignement inconscient et un clignement pour prendre des photos (plus lent que le clignement inconscient) pour éviter de prendre des photos de manière accidentelle.

Une rumeur émanant de représentants Google laisse entendre que les Google Glass permettraient aussi aux utilisateurs iPhone d'accéder à leur messagerie et à leur navigation. Si elle est vérifiée, ce sera un plus que la firme apportera puisqu'au départ, pour pouvoir le réaliser, les utilisateurs des Google Glass devaient au préalable disposer d'un téléphone Android et avoir l'application Glass Companion installée dessus.

Commentez la news de Stéphane Le Calme en ligne : [Lien 64](#)



### Comment manipuler la base de registre Windows en Perl ?

Le but de cet article est de vous montrer comment manipuler la base de registre en Perl : lecture ou création des clés/valeurs.

#### 1. Introduction

La base de registre (BDR) est une base de données utilisée par le système d'exploitation Windows. Elle contient les données de configuration des logiciels installés et également du système Windows lui-même. Les données sont faciles d'accès et la manipulation de cette base est faisable via l'outil Windows regedit. Il est possible de la manipuler via un programme écrit en VB, Perl ou autres. Cet article vous donne la possibilité de le faire en Perl.

Pour en savoir plus sur les bases de registre, n'hésitez pas à lire la FAQ Windows ([Lien 65](#)) sur developpez.com.

La manipulation de la base de registre Windows est très simple sous Perl. Il existe des modules bien écrits qui nous facilitent la vie. Il suffit de bien lire la documentation de ces modules afin de bien les utiliser.

Dans cet article, nous utiliserons le module Win32::TieRegistry et divers exemples : [Lien 66](#).

Dans la base de registre, les données sont organisées dans des HKEY (soit « poignée de clés »). Les données sont stockées dans des « ruches » qui sont des blocs, éventuellement chaînés de 4096 caractères.

Ces HKEY sont au nombre de cinq, leurs noms sont explicites :

- HKEY\_CLASSES\_ROOT ;
- HKEY\_CURRENT\_USER ;
- HKEY\_LOCAL\_MACHINE ;
- HKEY\_USERS ;
- HKEY\_CURRENT\_CONFIG.

Les programmes de cet article vont manipuler les données. Il suffit de connaître le chemin de l'information pour y accéder et/ou les manipuler, c'est ce qui est bien souvent le moins évident !

#### 2. Prérequis

Les programmes que nous utilisons dans cet article manipulent la base de registre. Par sécurité, nous vous recommandons de faire une sauvegarde préalable de cette dernière : [Lien 67](#).

Pour sauvegarder votre base de registre, lancez la commande suivante : **regedit**. Une fois l'interface graphique ouverte, exportez la base.

Pour tester les programmes de cet article sous Windows Vista, Seven, 8... vous devez installer le module Win32::TieRegistry : [Lien 66](#).

Il est très important de lancer les programmes ou d'ouvrir la console DOS avec les droits d'administrateur, sinon il

vous sera impossible de lire entièrement la base de registre pour des raisons de sécurité et il se peut que vous ayez le message « *Le système n'a pas trouvé l'option d'environnement spécifiée* ».

Pour en savoir plus sur l'installation des modules en Perl, n'hésitez pas à lire cet article : installation des modules CPAN ([Lien 68](#)).

#### 3. Lire la base de registre

##### 3.1. Obtenir des informations sur l'installation d'ActivePerl

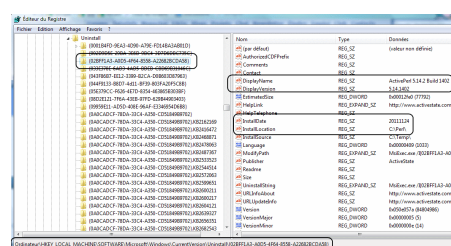
Si comme moi vous êtes sous Windows (Vista en ce qui me concerne) et utilisez ActivePerl, vous avez certaines informations d'ActivePerl dans votre base de registre suite à son installation que nous allons lister :

- lieu de location ;
- date d'installation ;
- version ;
- ...

Les informations d'installation des logiciels sous Windows se trouvent dans la base de registre à ce niveau : HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall.

Sous certains postes comme Windows Seven 64 bits, le chemin peut être : HKEY\_LOCAL\_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall. Pour en savoir plus, lisez cette note Microsoft : 32-bit and 64-bit Application Data in the registry ([Lien 69](#)).

Ensuite, dans le niveau du dessous, vous avez le nom du logiciel ou un code. Il va donc falloir parcourir la liste des noms et pour chacun d'eux, on recherche une clé se nommant « DisplayName ». Cela permet de s'assurer qu'il s'agit bien du logiciel recherché. Voici une capture d'écran montrant les informations à trouver et un programme pour rechercher les informations sur ActivePerl.



## ActivePerl : base de registre

```
#!/usr/bin/perl
use strict;
use warnings;
use Win32::TieRegistry;

my $nom_logiciel = 'ActivePerl';
if ( my $RefInfoPerl =
GetProductCode($nom_logiciel) ) {
    foreach my $cle ( keys %{$RefInfoPerl} ) {
        print "$cle : $RefInfoPerl->{$cle}\n";
    }
}

sub GetProductCode {
    my $Name      = shift;
    my $RegistryPath =
'HKEY_LOCAL_MACHINE/SOFTWARE/Microsoft/Windows/Cu
rrentVersion/Uninstall';
    my %InfoProduct;

    $Registry->Delimiter('/');

    my $SubKey = $Registry->{$RegistryPath} or die
"Lecture impossible : $RegistryPath\n$^E\n";

    foreach my $key ( keys %{$SubKey} ) {
        my $RefName = $SubKey->{$key}{DisplayName};

        if ( $RefName and $RefName =~ m{$Name}i ) {
            $InfoProduct{Code} = $key;
            if ( exists $SubKey->{$key}{DisplayName} )
            {
                $InfoProduct{DisplayName} = $SubKey-
>{$key}{DisplayName};
            }
            if ( exists $SubKey->{$key}
{InstallLocation} ) {
                $InfoProduct{InstallLocation} = $SubKey-
>{$key}{InstallLocation};
            }
            if ( exists $SubKey->{$key}{DisplayVersion}
) {
                $InfoProduct{DisplayVersion} = $SubKey-
>{$key}{DisplayVersion};
            }
            if ( exists $SubKey->{$key}{InstallDate} )
            {
                $InfoProduct{InstallDate} = $SubKey-
>{$key}{InstallDate};
            }
            last;
        }
    }

    return \%InfoProduct;
}
```

J'obtiens sur mon ordinateur le résultat suivant :

### Résultat

```
Code : {02BFF1A3-A0D5-4F64-8558-A22682BCDA58}/
DisplayVersion : 5.14.1402
InstallDate : 20111124
DisplayName : ActivePerl 5.14.2 Build 1402
InstallLocation : C:\Perl\
```

Le programme contient une procédure nommée « GetProductcode » qui nous permet de lister certaines informations. Le nom de logiciel que nous cherchons est

passé en argument. Dans le cas ci-dessus, c'est « ActivePerl », mais ça aurait pu être Adobe, Gimp...

En ligne 15, la variable contient le chemin dans la base de registre qui permet d'accéder à la plupart des logiciels installés. Les données récupérées sont stockées dans un hash qui est renvoyé par la procédure.

En ligne 18, nous précisons que le délimiteur est un slash « / ».

La ligne 20 permet de récupérer la base de registre au niveau de l'arborescence voulue. Ensuite, on parcourt ce hash pour extraire ce dont on a besoin.

N.B. La variable est une référence de hash.


En ligne 23, on récupère la valeur de la clé « DisplayName », ce qui nous permet de trouver réellement notre logiciel. Une fois trouvé, nous récupérons les valeurs des clés « InstallLocation », « DisplayVersion », et « InstallDate ». Exemple :

```
if ( exists $SubKey->{$key}
{InstallLocation} ) {
    $InfoProduct{InstallLocation} = $SubKey-
>{$key}{InstallLocation};
}
```

Comme vous pouvez le constater, c'est très simple. Le plus compliqué, c'est de connaître un peu l'arborescence de la base de registre pour savoir où chercher. Nous aurions pu écrire notre code différemment. Si vous voulez en savoir plus, lisez la documentation du module Win32::TieRegistry : [Lien 66](#).

## 4. Modifier la base de registre

Le but est maintenant de modifier notre base de registre. Pour cela, nous allons définir un exemple simple à réaliser. Nous avons des fichiers d'extension « .dvp » et souhaitons que ces fichiers soient ouverts systématiquement (après un double-clic) via notre éditeur de texte Notepad++.

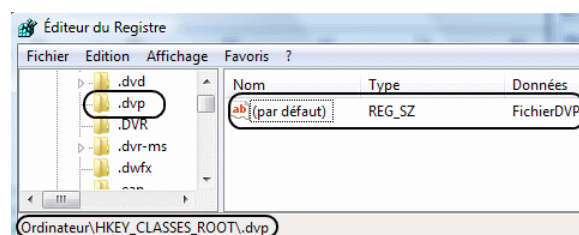
De plus, nous souhaitons que tous les fichiers « .dvp » de notre ordinateur aient une icône de notre choix qui est la suivante : . Nous allons voir comment le faire manuellement et en Perl.

Avant de faire quoi que ce soit, je vous recommande de faire une sauvegarde de votre base de registre (une exportation) !

### 4.1. Manuellement

Nous souhaitons travailler sur les extensions de type « .dvp ». Créons une clé **.dvp** dans l'HKEY « HKEY\_CLASSES\_ROOT ». À l'intérieur, attribuons à la clé « par défaut » une valeur de notre choix : **FichierDVP**.

Il est important de choisir un nom sans espace.



Pour créer la clé .dvp, vous faites un clic droit sur

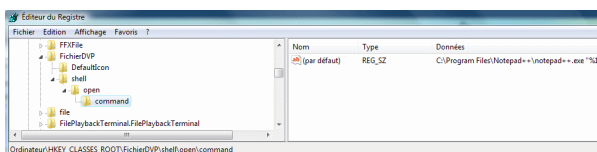
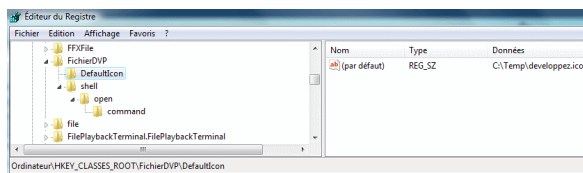
HKEY\_CLASSES\_ROOT, puis Nouveau → Clé.

Maintenant, il faut créer une clé du même nom choisi ci-dessus « FichierDVP » dans laquelle on crée deux clés :

- Une sous-clé « DefaultIcon ». Sa valeur par défaut aura le nom du fichier **developpez.ico** (chemin complet), c'est cette icône qui sera affichée dans l'explorateur Windows pour tous les fichiers d'extension .dvp. Pour l'article, l'icône se trouve dans le répertoire Temp : C:\Temp\developpez.ico. Nous aurions pu indiquer le nom d'un fichier .exe ou .dll suivi d'une virgule et du rang de l'icône voulue dans ce fichier (0, 1, 2... ) ;
- Une sous-clé « shell » pour action dans laquelle on crée une sous-clé « open », puis à l'intérieur une sous-clé « command ». Sa valeur par défaut contient la commande à exécuter (Notepad++). Dans la commande, on précisera l'argument %1 qui contient le nom du fichier. Pensez aux guillemets afin d'éviter les mauvaises surprises des noms de fichiers avec espaces.  
On aura donc shell → open → command et « C:\Program Files\Notepad++\notepad++.exe "%1" » en commande.

À ce stade, tout double-clic sur un fichier d'extension « .dvp » lancera Notepad++ avec en argument le nom du fichier. Le fichier sera ainsi ouvert dans Notepad++.

Pour voir vos fichiers .dvp avec l'icône dvp, vous devez redémarrer votre ordinateur ou au moins, fermer et relancer votre session.



Voyons maintenant comment faire cette modification de la base de registre en Perl.

## 4.2. Programme Perl

### Programme de création des clés

```
#!/usr/bin/perl
use strict;
use warnings;
use Win32::TieRegistry;

$Registry->Delimiter('/');
my $NomExtension = 'FichierDVP';

# Suppression des clés si présentes
$Registry->FastDelete(1);
delete $Registry->{"HKEY_CLASSES_ROOT/
$NomExtension/DefaultIcon/"};
delete $Registry->{"HKEY_CLASSES_ROOT/
$NomExtension/shell/open/command/"};
delete $Registry->{"HKEY_CLASSES_ROOT/
```

```
$NomExtension/shell/open/"};
delete $Registry->{"HKEY_CLASSES_ROOT/
$NomExtension/shell/"};
delete $Registry->{"HKEY_CLASSES_ROOT/
$NomExtension/"};
delete $Registry->{"HKEY_CLASSES_ROOT/.dvp/"};

# Création des nouvelles clés
# Clé .dvp
$Registry->{"HKEY_CLASSES_ROOT/.dvp/"} = { '/' =>
$NomExtension };

# Clé FichierDVP
$Registry->{"HKEY_CLASSES_ROOT/$NomExtension/"} =
{
    'DefaultIcon' => {
        '/' => 'C:\Temp\developpez.ico',
    },
    'shell' => {
        'open' => {
            'command' => {
                '/' => 'C:\Program Files\Notepad+
+\notepad++.exe "%1"'
            }
        },
    }
};
```

La ligne 6 permet de définir le délimiteur.

La ligne 7 définit la variable qui contient le nom de notre nouvelle extension.

### 4.2.1. Suppression des clés

De la ligne 10 à 16, on supprime les clés au cas où elles existaient. La méthode **FastDelete** permet d'économiser de la mémoire pendant la suppression des clés. En fait, si la méthode est appelée avec la valeur 1. La suppression des clés dans le hash se fait sans aucune copie. La mémoire n'est pas plus sollicitée. En appelant la méthode avec la valeur 0, la méthode **delete** retourne une copie de la portion de hash (une référence de hash) qui est supprimée.

```
$ref_copie = delete $Registry-
->{"HKEY_CLASSES_ROOT/.dvp/"};
```

0 : \$ref\_copie = référence de hash (si succès)

1 : \$ref\_copie = 1 (si succès)

Les lignes 11 à 16 ont un ordre à respecter. Une clé ne sera pas supprimée si cette dernière contient des sous-clés. Il faut donc supprimer en premier les sous-clés les plus basses dans l'arborescence. Ainsi, sachant que nous avons l'arborescence suivante : HKEY\_CLASSES\_ROOT → FichierDVP → DefaultIcon et HKEY\_CLASSES\_ROOT → FichierDVP → shell → open → command. Commençons par supprimer les clés DefaultIcon, command, open, shell et FichierDVP.

### 4.2.2. Création des clés

À partir de la ligne 20, nous créons nos clés. C'est assez simple, il suffit de passer à la variable **\$Registry** notre clé à créer **\$Registry->{"HKEY\_CLASSES\_ROOT/\$NomExtension/"}** puis on lui affecte une **référence de hash** qui contient toutes les sous-clés et valeurs. Pour renseigner les clés « par défaut », la clé du hash est un slash « / » ou une chaîne vide « ».

#### Valeur par défaut

```
'DefaultIcon' => {  
    '/' => 'C:\Temp\developpez.ico',  
},
```

ou

#### Valeur par défaut

```
'DefaultIcon' => {  
    '' => 'C:\Temp\developpez.ico',  
},
```

Vous avez maintenant toutes les clés en main pour vous

amuser avec les bases de registre.

#### **5. Conclusion**

Cet article vous a montré les méthodes simples pour manipuler une base de registre Windows en utilisant le module Win32::TieRegistry : [Lien 66](#).

Vous pouvez télécharger les sources de cet article : [Lien 70](#).

Je vous recommande de lire la documentation officielle du module afin d'en apprendre plus.

*Retrouvez l'article de Djibril en ligne : [Lien 71](#)*

### Présentation de LibreOffice

Ce chapitre est basé sur le Chapitre 1 du « Getting Started with OpenOffice.org ».

#### 1. Qu'est-ce que LibreOffice ?

LibreOffice est une suite bureautique complète et disponible gratuitement. Son format de fichier natif est OpenDocument, un format standard ouvert qui est adopté par les gouvernements à travers le monde comme format de fichier requis pour la publication des documents. LibreOffice peut aussi ouvrir et enregistrer les documents dans de nombreux autres formats, incluant ceux utilisés par plusieurs versions de Microsoft Office.

LibreOffice est composé des différents programmes.

##### 1.1. Writer (traitement de texte)

Writer est un outil aux fonctionnalités complètes permettant de créer des lettres, des livres, des rapports, des journaux, des brochures et autres documents. Vous pouvez insérer des images et des objets venant des autres composants dans les documents Writer. Writer peut exporter les fichiers au format HTML, XHTML, XML, Portable Document Format (PDF) Adobe et plusieurs versions des formats de fichier de Microsoft Word. Il se connecte également à votre client de messagerie.

##### 1.2. Calc (classeur)

Calc possède tous les outils d'analyse, de diagrammes et d'aide à la décision que l'on peut attendre des classeurs actuels. Il inclut entre autres près de 300 fonctions financières, statistiques et mathématiques. Le gestionnaire de scénarios fournit une analyse « et si ? ». Calc génère des diagrammes 3-D et 2-D qui peuvent être intégrés dans d'autres documents LibreOffice. Vous pouvez également ouvrir et travailler avec les classeurs Excel et les enregistrer au format Excel. Calc peut exporter les classeurs au format PDF Adobe et HTML.

##### 1.3. Impress (présentations)

Impress fournit tous les outils de présentations multimédias communs, tels que les effets spéciaux, les animations et des outils de dessin. Il intègre les outils graphiques avancés des composants Draw et Math. Les présentations peuvent être améliorées par l'outil de travail des polices et effets spéciaux de texte Fontwork ainsi que des clips vidéo et son. Impress est compatible avec le format de fichier PowerPoint et peut aussi enregistrer votre travail dans de nombreux formats graphiques, incluant Macromedia Flash (SWF).

##### 1.4. Draw (dessin vectoriel)

Draw est un outil de dessin vectoriel qui peut produire du simple diagramme au diagramme de flux 3-D. Sa fonction de connecteur permet de définir vos propres points de

connexion. Vous pouvez utiliser Draw pour créer des dessins à utiliser dans n'importe quel autre des composants LibreOffice et vous pouvez créer vos propres dessins et les ajouter à la Gallery. Draw peut importer les images dans la plupart des formats courants et les enregistrer dans plus de 20 formats incluant PNG, HTML, PDF, et Flash.

##### 1.5. Base (base de données)

Base fournit des outils pour le travail quotidien sur les bases de données à partir d'une interface simplifiée. Il peut créer et éditer des formulaires, des rapports, des requêtes, des tables, des vues et des relations ainsi que gérer une base de données connectée de la même façon que dans les autres applications populaires de base de données. Base fournit de nombreuses fonctionnalités telles que la possibilité d'analyser et d'éditer des relations à partir d'un éditeur graphique. Base incorpore HSQLDB comme moteur de base de données relationnelle par défaut. Il peut également utiliser dBASE, Microsoft Access, MySQL, ou Oracle, ou n'importe quelle base de données utilisant ODBC ou JDBC. Base fournit également la prise en charge d'un sous-ensemble ANSI-92 SQL.

##### 1.6. Math (éditeur de formule)

Math est l'éditeur de formule ou d'équation LibreOffice. Vous pouvez l'utiliser pour créer des équations complexes qui incluent des symboles ou des caractères non disponibles dans l'ensemble de caractères standards. Alors qu'il est habituel de créer des formules dans d'autres documents tels que Writer et Impress, Math peut aussi fonctionner comme un outil autonome. Vous pouvez enregistrer les formules dans le format standard Mathematical Markup Language (MathML) pour l'inclure dans les pages web et autres documents non créés par LibreOffice.

#### 2. Les avantages de LibreOffice

Voici quelques-uns des avantages de LibreOffice par rapport aux autres suites bureautiques :

- **pas de coût de licence** : LibreOffice est gratuit pour tous tant pour la distribution que l'utilisation. Beaucoup des fonctionnalités qui sont disponibles pour un coût supplémentaire dans d'autres suites sous forme d'extensions (comme l'export PDF) sont disponibles par défaut dans LibreOffice. Il n'y a pas de coûts cachés maintenant ou dans le futur ;
- **open source** : vous pouvez distribuer, copier et modifier le logiciel autant que vous le souhaitez dans le respect de la licence open source de LibreOffice ;



- **multiplateforme** : LibreOffice fonctionne sur plusieurs architectures et de multiples systèmes d'exploitation tels que Microsoft Windows, Mac OS X et Linux ;
- **prise en charge des langues étendue** : l'interface utilisateur de LibreOffice est disponible dans près de 50 langues et LibreOffice fournit des dictionnaires d'orthographe, des synonymes et de coupure des mots dans plus de 80 langues et dialectes. LibreOffice fournit également la prise en charge à la fois des scripts complexes Complex Text Layout (CTL) et les scripts de gauche à droite Right to Left (RTL) (tels qu'urdu, hébreu, et arabe) ;
- **interface utilisateur homogène** : tous les composants ont une apparence similaire les rendant faciles à utiliser et acquérir ;
- **intégration** : les composants de LibreOffice sont bien intégrés les uns avec les autres ;
- **granularité** : habituellement, lorsque vous modifiez une option, cela affecte tous les composants. Cependant, les options LibreOffice peuvent être paramétrées au niveau du composant et même au niveau du document ;
- **compatibilité de fichiers** : en plus du format natif OpenDocument, LibreOffice inclut des possibilités d'export PDF et Flash, ainsi que la prise en charge à l'ouverture et l'enregistrement d'un grand nombre de formats communs comprenant Microsoft Office, HTML, XML, WordPerfect et Lotus 1-2-3. Et à travers une extension fournie par défaut : la capacité d'importer et d'éditer des fichiers PDF et de créer des fichiers hybrides PDF/ODF ;
- **pas de verrouillage du vendeur** : LibreOffice utilise OpenDocument, un format de fichier XML (eXtensible Markup Language) développé comme un standard de l'industrie par OASIS (Organization for the Advancement of Structured Information Standards). Ces fichiers peuvent facilement être décompressés et lus par n'importe quel éditeur de texte et leur structure de développement est ouverte et publiée ;
- **vous pouvez vous exprimer** : les améliorations, les correctifs et les dates de sorties sont fixés par la communauté. Vous pouvez joindre la communauté et avoir un impact sur le cours du développement du produit que vous utilisez ;
- tous les composants partagent le même dictionnaire orthographique commun ainsi que d'autres outils qui sont utilisés de façon homogène dans la suite. Par exemple, les outils de dessin sont disponibles dans Writer et on peut également les trouver dans Calc, de façon similaire, mais avec des fonctions avancées dans Draw et Impress,
- vous n'avez pas besoin de connaître quelle application a été utilisée pour créer un fichier particulier. Par exemple, vous pouvez ouvrir un fichier Draw à partir de Writer ;

Vous pouvez en lire davantage à propos de LibreOffice et The Document Foundation sur leurs sites web [Lien 72](#) et [Lien 73](#).

### 3. Besoins minimums

LibreOffice 3.x nécessite l'un des systèmes d'exploitation suivants :

- **Microsoft Windows** 2000 (Service Pack 4 ou supérieur), XP, Vista, ou 7 ;
- **GNU/Linux** Kernel version 2.6.18 et glibc2 version 2.5 ou supérieure est requise ;
- **Mac OS X** 10.4 (Tiger) ou supérieur.

Certaines fonctions de LibreOffice (assistants et le moteur de base de données HSQLDB) nécessitent que le Java Runtime Environment (JRE) 1.6.x ou ultérieur soit installé sur votre ordinateur. Cependant LibreOffice fonctionnera sans prise en charge Java, seules certaines fonctionnalités ne seront pas disponibles. Si vous avez un PC ancien et que vous n'avez pas besoin des fonctions nécessitant le JRE, vous pouvez le désactiver pour accélérer le chargement du programme.

Pour une liste détaillée des prérequis, voir le site LibreOffice, [Lien 74](#).

### 4. Comment obtenir le logiciel

Le logiciel peut être téléchargé à partir de cette adresse : [Lien 75](#). Vous pouvez également télécharger le logiciel à partir d'un client Peer-to-Peer tel que BitTorrent, à la même adresse.

Les utilisateurs Linux trouveront LibreOffice inclus dans plusieurs des dernières distributions Linux (Ubuntu en est l'un des exemples).

### 5. Comment installer le logiciel

Les informations sur l'installation et le paramétrage de LibreOffice sur les divers systèmes d'exploitation pris en charge sont données ici : [Lien 76](#)

### 6. Extensions et add-ons

Extensions et add-ons sont disponibles pour améliorer LibreOffice. Plusieurs extensions sont installées avec le programme et d'autres sont disponibles à partir du dépôt officiel des extensions, [Lien 77](#). Voir tutoriel le paragraphe dans le tutoriel d'installation : [Lien 78](#).

### 7. Comment obtenir de l'aide

Ce livre, les autres guides LibreOffice, le système d'aide en ligne et le système d'aide aux utilisateurs supposent que vous êtes familier avec votre ordinateur et ses fonctions basiques telles que démarrer un programme, ouvrir et enregistrer un fichier.

#### 7.1. Système d'aide

LibreOffice vient avec un système d'aide étendu. C'est votre première ligne de soutien dans l'utilisation de LibreOffice.

Pour afficher le système d'aide complet, appuyez sur *F1* ou sélectionnez **Aide de LibreOffice** à partir du menu Aide. Vous pouvez choisir d'activer les infobulles et les infoballons sous **Outils** → **Options** → **LibreOffice** → **Général**.

Si les infobulles sont activées, placer le pointeur de la souris sur n'importe quelle icône affichera une petite étiquette contenant une courte explication sur la fonction

de l'icône. Pour des explications plus détaillées, sélectionnez **Aide** → **Qu'est-ce que c'est ?** Et maintenez le pointeur au-dessus de l'icône.

## 7.2. Soutien gratuit en ligne

La communauté LibreOffice ne développe pas seulement le logiciel, mais fournit également une aide gratuite basée sur le volontariat. Les utilisateurs de LibreOffice peuvent bénéficier d'un support en ligne de la communauté à travers les listes de mail et les forums. Il y a également des sites web réalisés par des utilisateurs qui fournissent des tutoriels et de l'aide gratuite.

### Support LibreOffice gratuit

<b>FAQ</b>	Réponses à la Foire Aux Questions <a href="#">Lien 79</a>
<b>Documentation</b>	Guides Utilisateurs, how-to, et autre documentation. <a href="#">Lien 80</a>
<b>Mailing Lists</b>	Aide fournie gratuitement par des utilisateurs expérimentés. <a href="#">Lien 81</a>
<b>Support International</b>	Sites LibreOffice dans votre langue : <a href="#">Lien 82</a> International mailing listes internationales : <a href="#">Lien 83</a>
<b>Options d'accessibilité</b>	Information sur les options d'accessibilité : <a href="#">Lien 84</a>
<b>Support Mac</b>	Support pour l'installation et l'utilisation sur Mac OS X : <a href="#">Lien 85</a>

## 8. Démarrer LibreOffice

Le moyen le plus courant de lancer n'importe quel composant de LibreOffice est d'utiliser le menu système, le menu standard par lequel la plupart des applications sont démarrées. Sous Windows, il s'appelle le menu Démarrer. Sous GNOME, il s'appelle le menu Applications. Sous KDE, il est identifié par le logo KDE. Sous Mac OS X, c'est le menu Applications.

Lorsque LibreOffice a été installé sur votre ordinateur, dans la plupart des cas, une entrée de menu pour chaque composant a été ajoutée au menu système. Le nom exact et l'emplacement de ces entrées de menu dépendent du système d'exploitation et de l'interface utilisateur graphique.

### 8.1. Démarrer à partir d'un document existant

Vous pouvez démarrer LibreOffice en double-cliquant sur le nom de fichier d'un document LibreOffice dans un gestionnaire de document tel que Windows Explorer. Le composant LibreOffice correspondant démarrera et le document sera ouvert.

Note pour les utilisateurs Windows

Si vous avez associé les types de fichier Microsoft Office avec LibreOffice, alors lorsque vous double-cliquez sur un fichier \*.doc (Word), il s'ouvre dans Writer; \*.xls (Excel), il s'ouvre dans Calc et \*.ppt (PowerPoint), il s'ouvre dans Impress.

Si vous n'avez pas associé les types de fichier, alors lorsque vous double-cliquez sur un fichier Microsoft Word, il s'ouvre dans Word (si Word est installé sur votre ordinateur), un fichier Excel s'ouvre dans Excel et un fichier PowerPoint dans PowerPoint.

Vous pouvez utiliser une autre méthode pour ouvrir les

fichiers Microsoft Office dans LibreOffice et enregistrer dans ces formats à partir de LibreOffice. Voir « » pour plus d'information.

## 9. Utiliser le démarrage rapide sous Windows

Le Démarrage rapide est une icône qui est placée dans la barre d'état du système Windows au démarrage du système. Elle indique que LibreOffice a été chargé et est prêt à être utilisé (le démarrage rapide charge la bibliothèque \*.DLL nécessaire à LibreOffice, réduisant le temps de démarrage des composants LibreOffice). Si le démarrage rapide est désactivé, voir « » si vous voulez l'activer de nouveau.

### 9.1. Utiliser l'icône de démarrage rapide

Faites un clic avec le bouton droit sur l'icône de **Démarrage rapide** dans la barre d'état pour ouvrir une fenêtre de menu () à partir de laquelle vous pouvez ouvrir un nouveau document, ouvrir la boîte de dialogue Modèles et documents et la boîte de dialogue Documents, ou encore choisir un document existant à ouvrir. Vous pouvez également double-cliquer sur l'icône de **Démarrage rapide** pour afficher la boîte de dialogue Modèles et documents.

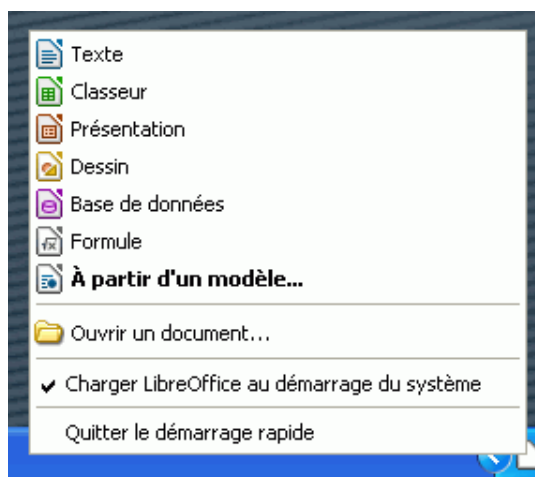


Figure : menu de démarrage rapide sous Windows.

### 9.2. Désactiver le démarrage rapide

Pour fermer le Démarrage rapide, faites un clic avec le bouton droit sur l'icône dans la barre d'état et cliquez ensuite sur **Quitter le démarrage rapide** dans le menu qui s'affiche. La prochaine fois que l'ordinateur sera démarré, le Démarrage rapide sera de nouveau lancé.

Pour empêcher LibreOffice de se charger pendant le démarrage du système, désélectionnez l'option **Charger LibreOffice au démarrage du système** dans la fenêtre de menu. Vous pouvez souhaiter faire cela si votre ordinateur n'a pas suffisamment de mémoire par exemple.

### 9.3. Réactiver le démarrage rapide

Si le démarrage rapide a été désactivé, vous pouvez le réactiver en sélectionnant la case à cocher **Charger LibreOffice au démarrage du système** sous **Outils** → **Options** → **LibreOffice** → **Mémoire**.

## 9.4. Utiliser le démarrage rapide sous Linux et Mac OS

### X

LibreOffice sous Linux a un démarrage rapide qui ressemble et agit comme celui décrit sous Windows ci-dessus (la case à cocher sur l'onglet Mémoire est appelée **Activer le démarrage rapide**).

Mac OS X fournit une fonctionnalité similaire à partir du menu du dock.

## 10. Parties de la fenêtre principale

La fenêtre principale est similaire dans chaque composant de LibreOffice, même si quelques détails varient. Reportez-vous aux chapitres à propos de Writer, Calc, Draw et Impress dans ce guide pour une description de ces détails.

Les fonctionnalités communes incluent la barre de menu, la barre d'outils standard et la barre de formatage en haut de la fenêtre ainsi que la barre d'état tout en bas.

### 10.1. Barre de menus

La barre de menus est située en haut de la fenêtre LibreOffice, juste en dessous de la barre de titres. Lorsque vous choisissez l'un des menus listés ci-dessous, un sous-menu affiche des commandes.

- **Fichier** contient les commandes qui s'appliquent au document entier telles que Ouvrir, Enregistrer, Exporter comme PDF ;
- **Édition** contient les commandes pour l'édition du document, telles que Annuler : xxx (où xxx est la commande à annuler) et Rechercher & Remplacer. Il contient aussi les commandes de copier, couper et coller ainsi que le collage spécial des parties sélectionnées du document ;
- **Affichage** contient les commandes pour contrôler l'affichage du document telles que le Zoom ou la mise en page Web ;
- **Insertion** contient des commandes pour l'insertion d'éléments dans votre document telles que l'en-tête, le pied de page et les images ;
- **Format** contient les commandes telles que Styles et formatage ou AutoCorrection, pour le formatage de la mise en page du document ;
- **Tableau** affiche toutes les commandes pour insérer et éditer un tableau dans un document texte ;
- **Outils** contient les fonctions telles que Orthographe et grammaire, Personnaliser, et les Options ;
- **Fenêtre** contient les commandes de la fenêtre d'affichage ;
- **Aide** contient les liens vers l'aide en ligne de LibreOffice, Qu'est-ce que c'est ? et des informations à propos du programme. Voir « ».

### 10.2. Barre d'outils

LibreOffice a plusieurs types de barre d'outils : ancrée (fixée en place), flottante et détachable. Les barres ancrées peuvent être déplacées à différents endroits ou rendues flottantes et les barres flottantes peuvent être ancrées.

La barre d'outils ancrée en haut, juste sous la barre de menu est appelée barre *Standard*. Elle est identique à travers les applications LibreOffice.

La seconde barre d'outils en haut est la barre Formatage. Elle est sensible au contexte ; c'est-à-dire qu'elle affiche les outils correspondant à la position active du curseur ou à la sélection. Par exemple lorsque le curseur est dans une image, la barre Formatage fournit des outils de formatage des images ; lorsque le curseur est dans un texte, les outils sont ceux du formatage de texte.

### 10.3. Afficher ou masquer les barres d'outils

Pour afficher ou masquer les barres d'outils, choisissez **Affichage** → **Barre d'outils**, puis cliquez sur le nom de la barre d'outils dans la liste. Une barre d'outils active affiche une coche en face de son nom. Les barres d'outils détachables ne sont pas listées dans le menu Affichage.

#### 10.4. Sous-menu et barre détachable

Les icônes de la barre d'outils avec un petit triangle sur la droite afficheront un sous-menu, des barres d'outils détachables et d'autres moyens de sélectionner les objets, en fonction de l'icône.

La page affiche une barre d'outils détachable à partir de la barre d'outils *Dessin*.

Les barres d'outils détachables peuvent être flottantes ou ancrées le long d'un bord de l'écran ou dans l'une des zones de barres existantes. Pour déplacer une barre d'outils détachable flottante, glissez-la par la barre de titre, comme affiché dans la .

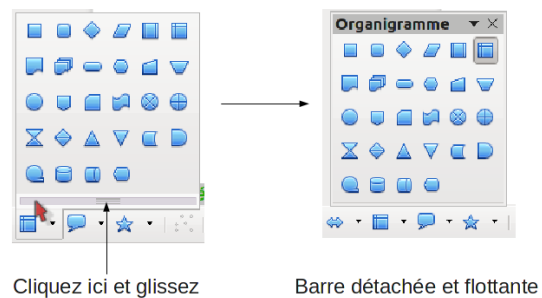


Figure : exemple de barres détachables.

### 10.5. Déplacer les barres d'outils

Pour déplacer une barre d'outils ancrée, placez le pointeur de la souris sur la poignée de la barre d'outils (la petite barre verticale à la gauche de la barre d'outils), maintenez le bouton gauche de la souris, glissez la barre d'outils à son nouvel emplacement, puis relâchez le bouton de la souris (). Pour déplacer une barre d'outils flottante, cliquez sur le titre de la barre et glissez-la à son nouvel emplacement (Figure ).

#### Poignées des barres ancrées

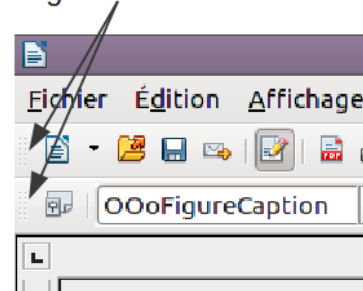


Figure : déplacer une barre ancrée.

Barre de titre de la barre flottante



Figure : déplacer une barre flottante.

### 10.6. Barres d'outils flottantes

LibreOffice comprend plusieurs barres d'outils supplémentaires sensibles au contexte, qui, par défaut, apparaissent comme des barres d'outils flottantes en réponse à la position actuelle du curseur ou à la sélection. Par exemple, lorsque le curseur est dans un tableau, une barre d'outils flottante *Tableau* apparaît et lorsque le curseur est dans une liste numérotée ou à puces, la barre d'outils *Puces et numérotations* apparaît. Vous pouvez ancrer ces barres d'outils en haut, en bas ou sur le côté de la fenêtre, si vous le souhaitez (voir « » ci-dessus).

### 10.7. Ancrer/rendre flottantes les fenêtres et barres d'outils

Les barres d'outils et certaines fenêtres, telles que le Navigateur et la fenêtre Styles et formatage sont ancrables. Vous pouvez les déplacer, les redimensionner ou les ancrer à un bord.

Pour ancrer une fenêtre ou une barre d'outils, maintenez la touche Ctrl et double-cliquez sur le cadre de la fenêtre flottant (ou sur un espace libre près des icônes en haut de la fenêtre flottante) afin de l'ancrer à sa dernière position.

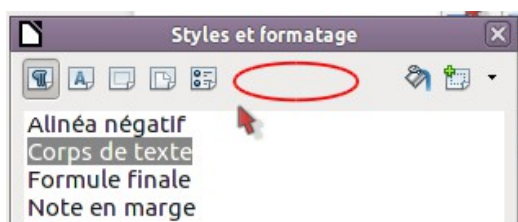


Figure : Ctrl+clic pour ancrer ou désancrer.

Pour désancrer une fenêtre, maintenez la touche Ctrl et double-cliquez sur le cadre (ou un espace libre près des icônes en haut) de la fenêtre ancrée.

La combinaison de touches Ctrl+Maj+F10 permet aussi d'ancrer et désancrer.

### 10.8. Personnaliser les barres d'outils

Vous pouvez personnaliser les barres d'outils de différentes façons, en choisissant quelles icônes sont visibles et en verrouillant la position d'une barre d'outils ancrée. Vous pouvez également ajouter des icônes et créer une nouvelle barre d'outils, comme décrit au chapitre 14.

Pour accéder aux options de personnalisation de la barre d'outils, faites un clic droit sur la barre pour afficher le menu, ou utilisez la flèche vers le bas à la fin de la barre d'outils ou sur sa barre de titre.

Options de personnalisation des barres



Figure : personnaliser les barres d'outils.

Pour afficher ou masquer les icônes définies pour la barre d'outils sélectionnée, choisissez **Boutons visibles** à partir du menu. Les icônes visibles sont indiquées par un cadre autour de l'icône. Cliquez sur l'icône pour la sélectionner ou la désélectionner.

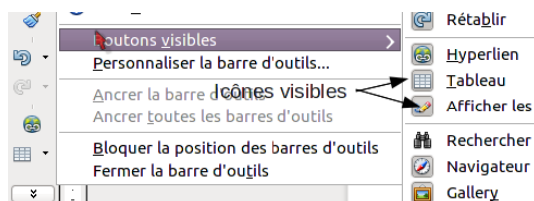


Figure : sélection des icônes visibles.

### 10.9. Menus par clic droit (contextuel)

Vous pouvez accéder rapidement à plusieurs fonctions de menu en faisant un clic avec le bouton droit sur un paragraphe, une image ou tout autre objet. Un menu contextuel s'affichera. Souvent, le menu contextuel est le moyen le plus rapide et le plus facile pour atteindre une fonction. Si vous n'êtes pas sûr de l'emplacement d'une fonction dans les menus ou barre d'outils, vous la trouverez souvent sous le clic droit.

### 10.10. Barre d'état

La barre d'état est située en bas de l'espace de travail. Elle fournit des informations sur le document et un moyen facile pour modifier rapidement certaines fonctions. Elle est similaire dans Writer, Calc, Impress et Draw bien que chaque composant inclue des éléments spécifiques à ce composant.

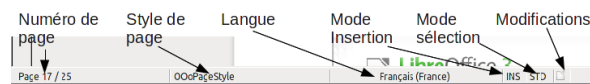


Figure : partie gauche de la barre d'état de Writer.

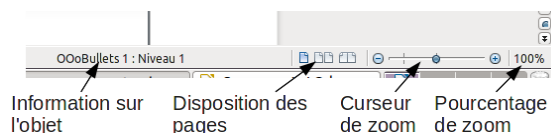




Figure : partie droite de la barre d'état.

Les éléments de la barre de statut communs sont décrits ci-dessous.

Numéro de page, de feuille ou de diapo : affiche le numéro de page, de feuille ou de diapo actif ou le nombre total de pages, feuilles ou diapos dans le document. Double-cliquez sur ce champ pour ouvrir le Navigateur. Les autres usages de ce champ dépendent du composant.

Style de page ou mise en page de diapo : affiche le style de page actif ou la mise en page de diapo active. Pour éditer le style de page ou la mise en page actifs, double-cliquez sur ce champ.

Modifications non enregistrées : une icône  apparaît ici si les modifications apportées au document n'ont pas été enregistrées.

Signature numérique : si le document a été signé numériquement, une icône  apparaît ici. Vous pouvez double-cliquer sur l'icône pour afficher le certificat.

Information sur l'objet : affiche des informations en



fonction de la position du curseur ou de l'élément sélectionné dans le document. Double-cliquer sur cette zone affiche habituellement la boîte de dialogue correspondante.

Zoom et pourcentages : pour modifier la vue, glissez le curseur de zoom ou cliquez sur les signes + et -, ou faites un clic avec le bouton droit sur le pourcentage de niveau de zoom pour afficher une liste de valeurs d'agrandissement pour faire votre choix.

Double-cliquer sur le niveau de pourcentage de zoom ouvre la boîte de dialogue **Zoom et disposition des pages**.

### 11. Comment s'appellent toutes ces choses ?

Les termes utilisés dans LibreOffice pour la plus grande partie de l'interface utilisateur (la partie du programme que vous voyez et utilisez, au contraire du code qui la fait fonctionner) sont les mêmes que pour la plupart des autres programmes.

Une *boîte de dialogue* est un type de fenêtre spécial. Son propos est de vous informer de quelque chose ou vous demander une saisie ou les deux. Elle fournit des contrôles pour vous permettre de spécifier comment mener une action. Les noms techniques des contrôles courants sont affichés dans la ; la zone de liste n'est pas montrée (à partir de laquelle vous sélectionnez un élément). Dans la plupart des cas, nous n'utilisons pas les termes techniques dans ce livre, mais il est utile de les connaître parce que l'Aide et d'autres sources d'information les utilisent souvent.

Dans la plupart des cas, vous ne pouvez interagir qu'avec la boîte de dialogue (pas le document lui-même) tout le temps que la boîte de dialogue reste ouverte. Lorsque vous fermez la boîte de dialogue après utilisation (habituellement en cliquant sur **OK** ou sur un autre bouton qui enregistre vos modifications et ferme la boîte de dialogue), alors vous pouvez à nouveau travailler avec votre document.

Certaines boîtes de dialogue peuvent être laissées ouvertes alors que vous travaillez, vous pouvez alors aller et venir entre la boîte de dialogue et votre document. Un exemple de ce type est la boîte de dialogue Rechercher et remplacer.

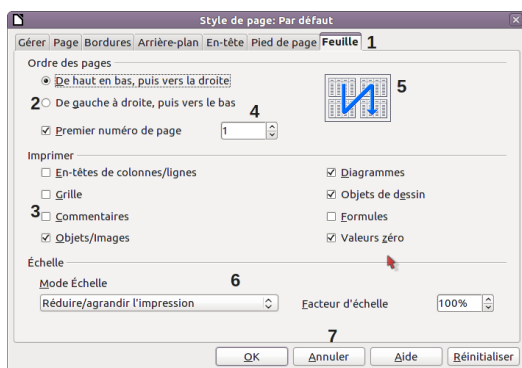


Figure : boîte de dialogue montrant des contrôles courants :

- 1=Onglet page (pas à proprement parler un contrôle)
- 2=Boutons radio (un seul peut être sélectionné)
- 3=Case à cocher (plusieurs peuvent être sélectionnées)
- 4=Compteur (cliquez sur les flèches haut et bas pour modifier le nombre affiché dans la zone de texte ou saisissez-le directement)
- 5=Vignette ou aperçu

6=Liste déroulante à partir de laquelle sélectionner un élément

7=Boutons à cliquer.

### 12. Démarrer un nouveau document

Vous pouvez créer un nouveau document vierge dans LibreOffice de plusieurs façons.

Lorsque LibreOffice est démarré, mais qu'aucun document n'est ouvert (par exemple si vous fermez tous les documents, mais laissez le programme s'exécuter), le Centre de démarrage est affiché. Cliquez sur l'une des icônes pour ouvrir un nouveau document de ce type, ou cliquez sur l'icône Modèle pour démarrer un nouveau document en utilisant un modèle.

Vous pouvez également créer un nouveau document de l'une des façons suivantes :

- utilisez **Fichier** → **Nouveau** et choisissez le type de document ;
- utilisez la flèche à côté du bouton **Nouveau** dans la barre d'outils principale. À partir du menu déroulant, sélectionnez le type de document à créer ;
- appuyez sur *Ctrl+N* sur le clavier ;
- utilisez **Fichier** → **Assistants** pour certains types de document spéciaux.

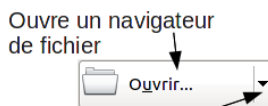
Si un document est déjà ouvert dans LibreOffice, le nouveau document s'ouvre dans une nouvelle fenêtre.



Figure : centre de démarrage LibreOffice.

### 13. Ouvrir un document existant

Lorsqu'aucun document n'est ouvert, le Centre de démarrage fournit une icône pour ouvrir un document existant ou choisir à partir d'une liste de documents récemment édités.



Ouvre la liste des documents récemment ouverts

Vous pouvez également ouvrir un document existant de l'une des façons suivantes :

- choisissez **Fichier** → **Ouvrir...** ;
- cliquez sur le bouton **Ouvrir** dans la barre d'outils principale ;
- appuyez sur *Ctrl+O* à partir du clavier.

Dans tous les cas, la boîte de dialogue Ouvrir apparaît. Sélectionnez le fichier souhaité puis cliquez sur **Ouvrir**. Si



un document est déjà ouvert dans LibreOffice, le second document s'ouvre dans une nouvelle fenêtre.

Dans la boîte de dialogue Ouvrir, vous pouvez réduire la liste de fichiers en sélectionnant le type de fichier que vous recherchez. Par exemple, si vous choisissez **documents texte** comme type de fichier, vous ne verrez que les documents que Writer peut ouvrir (incluant **.odt**, **.doc**, **.txt**) ; si vous choisissez **Classeur**, vous ne verrez que les **.ods**, **.xls**, et autres fichiers que Calc peut ouvrir.

Vous pouvez également ouvrir un document existant qui est dans un format OpenDocument en double-cliquant sur l'icône de fichier sur le bureau ou dans un gestionnaire de fichiers tel que Windows Explorer.

Si vous avez associé les formats de fichier Microsoft Office avec LibreOffice, vous pouvez également ouvrir ces fichiers en double-cliquant dessus.

Sous Microsoft Windows, vous pouvez utiliser les boîtes de dialogue Ouvrir et Enregistrer sous de LibreOffice ou celles fournies par Microsoft Windows. Voir « ».

#### **14. Enregistrer un document**

Pour enregistrer un nouveau document, exécutez l'une des actions suivantes :

- appuyez sur **Ctrl+S** ;
- choisissez **Fichier** → **Enregistrer** à partir de la barre de menu ;
- cliquez sur le bouton **Enregistrer** dans la barre d'outils principale.

Lorsque la boîte de dialogue Enregistrer sous apparaît, saisissez le nom du fichier, vérifiez le type de fichier (si nécessaire) et cliquez sur **Enregistrer**.

Pour enregistrer un document ouvert avec le nom de fichier actif, choisissez **Fichier** → **Enregistrer**. Cela écrasera le dernier état enregistré du fichier.

#### **15. Protection par mot de passe**

Pour protéger un document contre son ouverture sans mot de passe, utilisez l'option de saisie d'un mot de passe dans la boîte de dialogue Enregistrer sous.

1. Dans la boîte de dialogue Enregistrer sous, sélectionnez **Enregistrer avec mot de passe** puis cliquez sur **Enregistrer**. Vous recevrez une invite de saisie ( ) ;
2. Saisissez le même mot de passe dans les deux champs, puis cliquez sur **OK**. Si les mots de passe correspondent, le document est enregistré protégé par un mot de passe. Si les mots de passe ne correspondent pas, vous recevrez un message d'erreur. Fermez la boîte de dialogue du message pour retourner à la boîte de dialogue de définition du mot de passe et saisissez-le à nouveau.

LibreOffice utilise un mécanisme de chiffrement très fort qui rend pratiquement impossible la récupération du contenu du document au cas où vous perdriez le mot de passe.

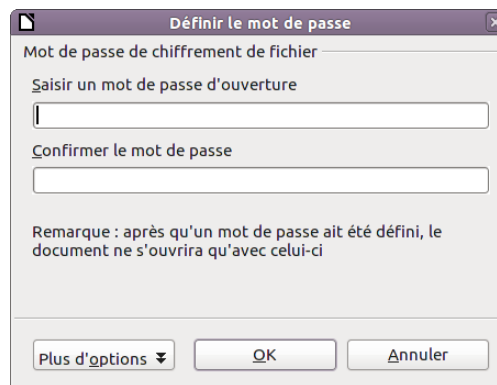


Figure : saisir un mot de passe pour le document.

Writer et Calc fournissent un second niveau de protection qui permet à un fichier d'être vu, mais non modifié sans mot de passe ; c'est-à-dire que le fichier s'ouvre en lecture seule.

Pour protéger un document contre les modifications uniquement :

1. Choisissez *Plus d'options* dans la boîte de dialogue *Définir un mot de passe* ;
2. Saisissez le mot de passe dans le champ *Saisir un mot de passe d'ouverture*. Répétez le mot de passe dans le champ *Confirmer le mot de passe*. Cliquez sur **OK**.

#### **16. Renommer et supprimer des fichiers**

Vous pouvez renommer ou supprimer des fichiers via les boîtes de dialogue de LibreOffice, comme vous pouvez le faire habituellement avec le gestionnaire de fichiers. Cependant vous ne pouvez pas copier ou coller à travers ces boîtes de dialogues.

#### **17. Ouvrir et Enregistrer sous**

Vous pouvez choisir d'utiliser les boîtes de dialogue Ouvrir et Enregistrer sous de LibreOffice ou celles fournies par votre système d'exploitation.

Pour voir ou modifier le type de boîtes de dialogue utilisé par LibreOffice :

1. choisissez **Outils** → **Options** → **LibreOffice** → **Général** ;
2. sélectionnez l'option **Utiliser les boîtes de dialogue LibreOffice**.

Cette section traite des boîtes de dialogue *Ouvrir* et *Enregistrer sous* de LibreOffice, la montre la boîte de dialogue *Enregistrer sous* ; la boîte de dialogue *Ouvrir* est similaire.

Les trois boutons en haut à droite des boîtes de dialogue *Ouvrir* et *Enregistrer sous*, de gauche à droite :

- **au répertoire supérieur** remonte d'un niveau dans la hiérarchie du répertoire. Cliquez et maintenez ce bouton pendant une seconde pour afficher une liste des dossiers de niveau supérieur ; pour vous rendre dans l'un de ces dossiers, déplacez le curseur de la souris sur son nom et relâchez le bouton ;
- **créer un nouveau répertoire** ;
- **répertoire par défaut**.

Pour les documents LibreOffice qui ont été enregistrés avec plus d'une version, utilisez la liste déroulante **Version**

pour sélectionner la version à ouvrir en mode lecture seule. Pour les documents Microsoft Office, seule la version actuelle peut être ouverte.

Utilisez le champ **Type de fichier** pour spécifier le type de fichier à ouvrir ou le format du fichier à enregistrer.

L'option **Lecture seule** dans la boîte de dialogue Ouvrir ouvre le fichier en lecture ou impression uniquement. En conséquence, la plupart des barres d'outils disparaissent et la plupart des options des menus sont inactives. Un bouton **Éditer le fichier** est affiché dans la barre d'outils Standard pour ouvrir le fichier pour édition.

Vous pouvez ouvrir un fichier à partir du Web en saisissant une URL dans le champ **Nom de fichier** dans la boîte de dialogue *Ouvrir*.

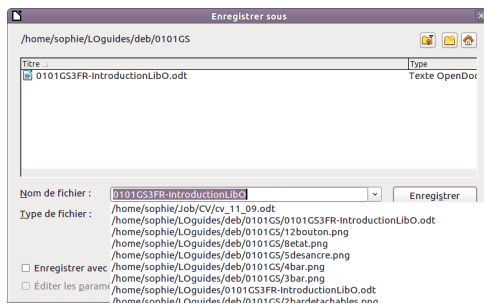



Figure : la boîte de dialogue Enregistrer sous LibreOffice.

## 18. Utiliser le Navigateur


Le Navigateur liste les objets contenus dans un document, regroupés dans des catégories. Par exemple, dans Writer, il affiche les titres, les tableaux, les cadres de texte, les commentaires, les images, les repères et d'autres éléments comme montrés dans la Figure . Dans Calc il affiche les feuilles, les noms des plages, les plages de base de données, les images, les objets de dessin et d'autres éléments. Dans Impress et Draw, il affiche les diapos, les images et encore d'autres éléments.

Pour ouvrir le Navigateur, cliquez sur l'icône  dans la barre d'outils *Standard* ou appuyez sur *F5*, ou choisissez **Affichage** → **Navigateur** dans la barre de menu.

Vous pouvez ancrer le Navigateur de chaque côté de la fenêtre principale de LibreOffice ou le laisser flottant (voir « »).

Cliquez sur le marqueur (ou le triangle +) de n'importe quelle catégorie pour afficher la liste des objets de cette catégorie.

Pour masquer la liste des catégories et afficher uniquement la barre d'outils en haut, cliquez sur l'icône

**Afficher/Masquer la zone de liste** . Cliquez de nouveau sur l'icône pour afficher la zone de liste.

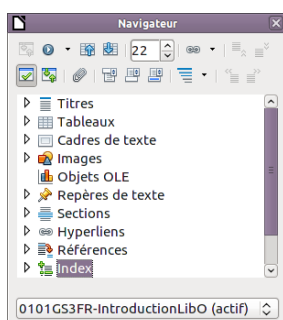



Figure : le Navigateur.


Le Navigateur fournit plusieurs moyens pratiques de se déplacer dans le document ou de trouver des éléments dans son contenu :

- lorsqu'une catégorie affiche la liste des objets qu'elle contient, double-cliquer sur un objet saute directement à l'emplacement de cet objet dans le document.

Les objets sont plus faciles à trouver si vous leur avez donné des noms lors de leur création au lieu de laisser celui par défaut de LibreOffice comme image1, image2, Tableau1, Tableau2 et ainsi de suite - ce qui peut ne pas correspondre à la position réelle de l'objet dans le document.

Si vous souhaitez seulement voir le contenu d'une certaine catégorie, mettez la catégorie en surbrillance et cliquez sur

l'icône **Affichage du contenu** . Jusqu'à ce que vous cliquiez de nouveau sur l'icône, seuls les objets contenus dans cette catégorie seront affichés :

- cliquez sur l'icône **Navigation**  (seconde icône à partir du haut gauche du Navigateur) pour afficher la barre d'outils *Navigation*. Ici, vous pouvez sélectionner l'une des catégories et utiliser les icônes **Précédent** et **Suivant** pour vous déplacer d'un élément à l'autre. C'est particulièrement utile pour trouver des éléments tels que les références et les index, qui peuvent être difficiles à voir.

Le nom des icônes (affiché dans l'infobulle) change pour refléter la catégorie sélectionnée ; par exemple, **Image suivante** ou **Référence suivante**.

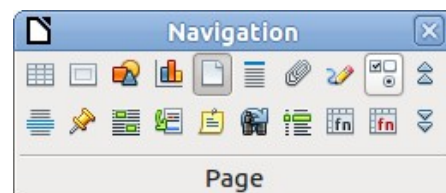



Figure : barre d'outils de navigation.

- pour sauter à une page donnée dans le document, saisissez son numéro dans le champ en haut du Navigateur.

Un peu d'expérience avec les autres icônes vous apprendra leur fonction. Quelques usages spécifiques au composant utilisé sont décrits dans les chapitres sur Writer et les autres composants concernés.

## 19. Annuler et refaire des modifications

Pour annuler les modifications les plus récentes appuyez

sur *Ctrl+Z*, ou cliquez sur l'icône **Annuler**  dans la barre d'outils *Standard* ou choisissez **Édition** → **Annuler** à partir de la barre des menus.

Le menu **Édition** affiche les dernières modifications qui peuvent être annulées (voir ci-dessous un exemple Writer).

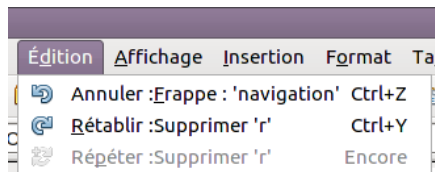


Figure : Édition → Annuler la dernière action.  
Cliquez sur le petit triangle à la droite de l'icône **Annuler** pour obtenir une liste de toutes les modifications qui peuvent être annulées. Vous pouvez sélectionner de multiples modifications et les annuler en une seule fois.

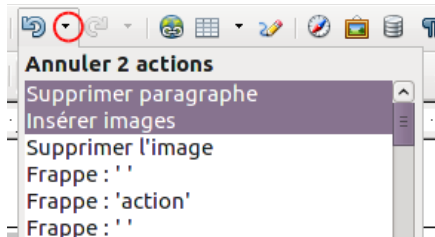



Figure : liste des actions qui peuvent être annulées.  
Après que des modifications ont été annulées, l'icône **Rétablir** devient active. Pour rétablir une modification, sélectionnez **Édition** → **Rétablir**, ou appuyez sur *Ctrl+Y* ou cliquez sur l'icône Rétablir . Comme pour Annuler, cliquez sur le triangle à la droite de la flèche pour obtenir une liste des modifications qui peuvent être appliquées de nouveau.

Pour modifier le nombre de modifications que LibreOffice peut enregistrer, choisissez **Outils** → **Options** → **LibreOffice** → **Mémoire** et dans la section Annuler, modifiez le **Nombre d'étapes**. Plus vous augmentez le nombre de modifications dont LibreOffice doit se rappeler et plus vous consommez de mémoire de l'ordinateur.

## 20. Fermer un document




Pour fermer un document, choisissez **Fichier** → **Fermer**. Vous pouvez également fermer un document en cliquant sur l'icône **Fermer** dans la fenêtre du document. Ce bouton ressemble à la X montrée . Elle peut être à un endroit différent en fonction de votre système d'exploitation.



Figure : icônes de fermeture.  
Si plus d'une fenêtre LibreOffice est ouverte, chaque fenêtre ressemble à celle montrée dans la . Fermer cette fenêtre laisse les autres fenêtres LibreOffice ouvertes. Si seulement une fenêtre LibreOffice est ouverte, elle

ressemble à l'image de droite de la . Remarquez la petite croix sous la grande. Cliquez sur la petite x ferme le document, mais laisse LibreOffice ouvert. Cliquez sur la grande X ferme LibreOffice complètement. Si le document n'a pas été enregistré depuis les dernières modifications, un message s'affiche. Choisissez si vous souhaitez enregistrer ou ignorer les modifications.

- **Enregistrer** : le document est enregistré puis fermé ;
- **Ignorer** : le document est fermé et toutes les modifications depuis le dernier enregistrement sont perdues ;
- **Annuler** : il ne se passe rien et vous retournez dans le document.

Ne pas enregistrer votre document peut entraîner la perte de modifications réalisées récemment, ou plus grave, du fichier entier.

## 21. Fermer LibreOffice

Pour fermer LibreOffice complètement, choisissez **Fichier** → **Quitter**, ou fermez le dernier document ouvert comme décrit dans « » ci-dessus.

Si tous les documents ont été enregistrés, LibreOffice se ferme immédiatement. Si un document a été modifié, mais non enregistré, un message d'avertissement apparaît. Suivez la procédure décrite dans « » pour enregistrer ou ignorer vos modifications.

## 22. Utiliser LibreOffice sur un Mac

Certaines combinaisons de touches et éléments de menu sont différents sur un Mac par rapport à ceux sous Linux ou Windows. Le tableau suivant donne les correspondances communes utilisées pour les instructions dans ce livre. Pour une liste plus détaillée, reportez-vous à l'aide.

Windows/Linux	Équivalent Mac	Effet
Menu <b>Outils</b> → <b>LibreOffice</b>	→	Accès aux options de paramétrage
<b>Options</b>	<b>Préférences</b>	Ouvre un menu contextuel
Clic avec le bouton droit	le <i>Ctrl+clic</i>	Utilisé avec d'autres touches
Ctrl (Contrôle)	<i>z (Commande)</i>	Ouvre le Navigateur
F5	<i>Maj+z+F5</i>	Ouvre la fenêtre Styles et formatage
F11	<i>z+T</i>	

*Retrouvez l'article de Ron Faile Jr, Jeremy Cartwright et Jean Hollis Weber traduit par Sophie Gautier en ligne : [Lien 86](#)*

### Interview de Clément « c1702 » Corde - Développeur de six jeux vidéo en amateur

Suite à un billet dans lequel c1702 présente ses projets de jeux vidéo, nous avons décidé d'en savoir plus et de prendre contact avec lui afin de savoir comment il en était arrivé jusque-là.

#### 1. Présentation



#### Pouvez-vous vous présenter en quelques mots ? Qui êtes-vous ?

Clément CORDE et pour mon âge, hmmm... Moi-même je préfère ne pas trop y penser...

#### Comment avez-vous appris la programmation ?

Quand j'étais gamin, le côté programmation m'intéressait, mais à l'époque il n'y avait que le BASIC. Mais bon, j'essayais quand même !

C'est à partir de l'IUT (en 1992) que j'ai vraiment commencé à m'y mettre, à avoir quelques bases, et à faire un peu de démos à côté en même temps.

#### Quel est votre parcours éducatif ?

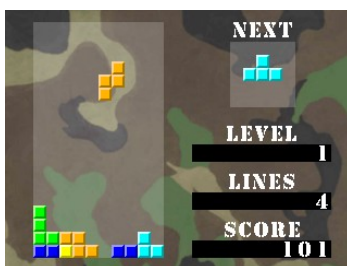
Eh bien un DUT informatique ! Après j'ai fait quelques autres trucs sans trop de rapport avec l'informatique, surtout pour reculer le moment d'aller faire le service militaire.

#### 2. Carrière

##### Quel est votre parcours professionnel ?

Comme 50 % des gens dans le milieu du jeu vidéo d'il y a quelques années, j'ai commencé chez Titus. C'est une boîte qui avait ses problèmes, mais qui avait aussi des bons points, comme embaucher des débutants.

Pour les boîtes qui ont suivi, ça gravitait autour d'une personne et il se trouve que je suis toujours en procès avec, donc je ne vais pas lui faire de pub.



#### Quel est votre emploi actuel ?

Je travaille dans la fonction publique. Jusqu'à il y a peu j'étais administrateur réseau.

#### Faites-vous toujours de la programmation dans votre travail ? Cela vous manque-t-il ?

Non, depuis que j'ai quitté le jeu vidéo, je ne programme plus au niveau pro. Et ça ne me manque pas vraiment : les plannings débiles avec deadlines intenables, le chef de projet neu-neu qui marque deux jours dans son planning alors qu'on vient de lui dire qu'il en faut trois et qui ne comprend pas deux jours plus tard que ce ne soit pas fini, les heures sup' à gogo non rémunérées... Non vraiment, ça ne me manque pas.

Surtout que dans le privé avec un BAC+2, aucun espoir d'évolution n'est possible.

#### 3. La programmation

##### Quels sont vos langages préférés ?

Je suis très « bas niveau », donc le C ! J'aimais beaucoup l'assembleur quand j'étais plus jeune (68000 et 80x86, et pseudo Z80 quand je faisais de la GameBoy), mais à l'heure actuelle, je trouve que GCC s'en sort plutôt bien et le C a l'énorme avantage d'être portable. En vrai.

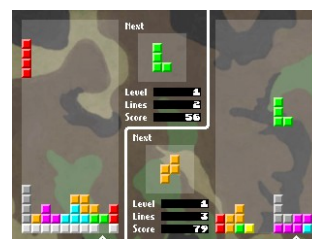
##### Quelles sont vos bibliothèques préférées ?

Je n'utilise que la SDL, qui elle aussi se trouve sur à peu près toutes les plateformes, donc c'est très pratique. Ensuite, j'aime bien réinventer la roue, donc je n'utilise pas de SDL\_xxx, même si je ne doute pas que certaines soient bien fichues. Ah oui, et j'ai récemment aussi réussi à utiliser ST-Sound de Léonard/Oxygène (la bibliothèque est super bien faite) avec la SDL, donc je l'utilise aussi, liée en statique.

Si j'avais le temps, j'aimerais bien voir un peu comment fonctionnent les cartes vidéo modernes, parce que dans les démos, on voit des trucs incroyables en quelques kilos.

##### Pourquoi avoir choisi le C avec la SDL ?

Le C, parce que je ne connais que ça en langage « évolué », que c'est assez bas niveau, portable, et que je n'avais pas envie de me remettre à l'ASM. La SDL, parce que quand j'ai repris la programmation, je voulais un accès à l'écran et qu'avec la SDL on a un pointeur dans le buffer vidéo en trois lignes (hors vérifications). Bonus, la SDL existe sur des tonnes de plateformes !





### Pourquoi vouloir utiliser directement le buffer ?

Pour le fun. Quand j'ai repris doucement la programmation, j'avais envie de refaire toutes les routines de base (sprites et compagnie). Il se trouve que finalement c'est plutôt rapide, mais à la base ce n'était pas du tout dans un but de performance.

## 4. Demoscene

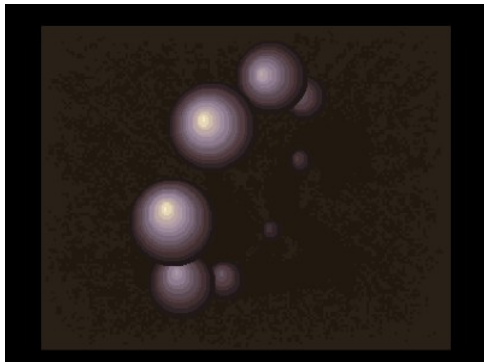
### Dans quels groupes avez-vous participé ?

J'ai uniquement participé à un groupe qui s'appelait « US ». Le nom qui est resté est « Universal Soldiers », mais le but était de changer de nom à chaque fois. Malheureusement, le groupe n'a pas duré très longtemps, et je n'ai pas gardé de contacts.

### Quelles démos avez-vous réalisées ?

À priori tout est là : [Lien 87](#)

(Moi je n'ai participé qu'à Unsatisfied et à Yul Brynner.)



### Est-ce de l'ASM ? Racontez-nous vos expériences, anecdotes avec ce langage. Coup de rush, démo finie à la va-vite ?

Oui, c'était de l'assembleur (pas le choix). On avait été à « The Party 4 » au Danemark. On avait fini (je crois) notre 40 kb sur place, mais sans plus que ça de précipitation.

### Atari ou Amiga ?

Dans les années 90, j'avais un ST. À l'époque je n'étais que joueur. Eh oui, il n'y avait pas de net, et se mettre au 68000 tout seul dans son coin en partant de rien paraissait complètement inaccessible.

Venant des 8 bits, Airball et Goldrunner m'ont beaucoup marqué !

Quand j'ai commencé à coder, sur PC, j'avais dans ma classe un certain Schmoovy-Schmoov de Melon Deziq, du coup j'ai assez vite acheté un Amiga 1200.

Franchement, je crois que quiconque a codé sur Amiga ne pourra qu'approuver : un 500 était bien supérieur à un ST. Sur le son ou la vidéo, il n'y a pas photo. Et puis le copper, quelle classe ! Sur ST, pour faire de l'overscan, il fallait si je ne m'abuse compter les cycles, switcher la fréquence de balayage au bon moment, etc. Bref, c'était déjà de la bidouille de très haut vol. Sur Amiga, il suffisait de faire sa petite copperlist, préciser que l'écran commence là, fait 384 pixels de large, et basta.

### Aimez-vous programmer pour l'Amiga ? Quels sont les outils et le langage ? Quelles sont les contraintes ? Continuez-vous à développer sur Amiga ?

J'aimais bien, oui !

À l'époque c'était assembleur obligatoire (ASM-One), parce qu'à 7 MHz, pas facile de tenir la frame autrement.

Non, je ne développe plus sur Amiga Classic. Par contre avec HunoPPC ([Lien 88](#)), on a porté mes jeux sur Amiga NG !



### Quelles sont vos démos préférées (oldschool & newschool) ?

Je n'ai pas suivi la « scène » depuis un bon moment. Apparemment ça a beaucoup évolué, mais c'est marrant de voir que certains sont encore dedans depuis tout ce temps ! Par exemple Chaos de Farbrausch, qui était à l'époque dans Sanity ! (Et qui était déjà très très bon à l'époque !) J'ai découvert récemment les démos et intros PC de Razor 1911, codées par Rez, et je trouve ça génial. C'est vraiment du oldschool moderne !

Je ne résiste pas à donner quelques liens :

- [Lien 89](#) (musique de 4mat, un vieux de la vieille également !)
- [Lien 90](#)
- [Lien 91](#)
- [Lien 92](#)

(Revenons au vif du sujet...)

## 5. La programmation de nos jours

### J'ai vu que vous avez fait un essai avec le C++, était-ce une volonté de « toucher » à un autre langage ? Quelles raisons vous ont motivé à tester le C++ ? Pourquoi avoir indiqué « mauvais C++ » ?

J'en avais fait un peu à une époque, et déjà je n'avais pas accroché. J'ai voulu essayer d'en refaire un peu (j'avais l'illusion que ça pourrait me servir) et mal m'en a pris. Dans l'outil dont tu parles, j'ai fait à peu près tout ce qu'il ne faut pas faire en ++. Si j'avais du temps, il faudrait que je repasse tout ça en C.

### Quels sont d'après vous les problèmes du C++ ?

Je n'ai pas grand-chose à dire à propos du C++, étant donné que je ne pratique pas du tout les langages « objet ». Je n'ai jamais réussi à me faire à la façon « objet » de penser, et j'ai l'impression qu'en C++ on passe son temps à réécrire sans arrêt les mêmes choses. Pour moi, ça donne un code plus lourd et je ne vois pas réellement de gain. Mais bon, je n'y connais rien.

### Avez-vous tenté un autre langage orienté objet ?

Non, je n'en ai pas eu l'occasion.

### Comment choisissez-vous un nouveau projet ? Quels sont les critères qui vous permettent de dire si tel projet est réalisable ou non ?

Un peu au feeling. Déjà s'il y a des maths, c'est non ! Ensuite sur des jeux 2D, mon humble expérience aidant, je vois à peu près ce que je vais être capable de faire ou pas. Tout en essayant quand même de faire des choses que je n'ai pas encore faites, sinon ce n'est pas drôle !

### Vous ne nous avez présenté que des projets qui ont abouti, mais avez-vous eu des projets qui n'ont pas abouti ? Si oui, pourquoi pensez-vous qu'ils n'aient pas abouti ? Si non, qu'est-ce qui selon vous te permet de



## réussir tous vos projets ?

Non, je n'ai pas vraiment de projet qui n'ait pas abouti depuis quelque temps. Mais il faut dire que j'évite de me lancer dans des choses infaisables.

### Quel processus utilisez-vous lors de vos projets ?

Je fais à ma façon, donc personnalisé ! :p En général, j'essaie d'avoir une base (très) solide avant d'avancer sur autre chose, ce qui m'évite d'avoir à déboguer au niveau « moteur » une fois que je suis dans le codage du « jeu ».

### Pour vous, quelles sont les qualités/prérequis nécessaires à programmeur pour réussir un petit projet ?

Que faut-il faire pour réussir au mieux un projet ?

Je crois que le tout c'est de ne pas se lancer dans des choses infaisables. Le truc classique, ce sont les gens qui commencent sans se rendre compte dans quoi ils se lancent. Par exemple « tiens, je vais faire un petit Mario... » Dans un Mario, il y a des graphismes enfantins, c'est sympa, tout ça. Sauf que ça reste un platformer, que c'est très compliqué et que donc ce n'est pas du tout par ça qu'il faut commencer ! Et on ne parle même pas du fait qu'il faut un éditeur de map !

Ensuite, bien garder en tête la règle des 90/90 ([Lien 93](#)), et rester motivé, parce que c'est toujours plus long que ce qu'on imagine.

Le conseil que je peux donner, c'est de commencer par de petits projets, terminables. Plusieurs petits projets menés de bout en bout apporteront plus qu'une grosse usine à gaz qui explosera de partout et qui ne sera jamais terminée.

## 6. Projets

### Est-ce que le Tetris est votre premier projet ou vous aviez déjà fait des petits « essais » avant ?

Celui-ci, je l'ai fait après l'Arkanoid, à la base comme un petit exemple pour le fils d'un collègue. Avant l'Arkanoid, j'avais recodé plein de petits effets de demomaker pour me remettre un peu dans le bain. Et ce n'est pas le premier Tetris que je faisais dans ma vie. (Même si depuis, avec Tetris Friends, je me suis aperçu qu'il faudrait rajouter quelques fonctionnalités.)

### Si j'ai bien compris, vous codez tout seul, mais d'où viennent les musiques et les images ? Est-ce que vous récupérez des images libres ou est-ce que vous les faites vous-même ?

Les images, je les trouve sur le net. Pour les sprites, il y a une bonne ressource : [Lien 94](#) (il y a aussi du son sur un site frère). Ou alors je les rippe moi-même quand ce n'est pas trop compliqué.

Je ne suis pas graphiste, donc j'utilise des graphismes existants (la remise en planche, les recalages, etc. prennent déjà pas mal de temps). Le tout c'est de le préciser et de ne pas tenter de faire d'argent avec. À partir de là, je n'ai jamais été embêté.

Le son (qui n'est pas mon fort), je récupère les sons que je peux (libres si possible) et je les bidouille avec Audacity. Et récemment, j'ai réussi à utiliser ST-Sound de Leonard/OXG ([Lien 95](#)) avec la SDL, ce qui m'a donné des idées.



### Vous semblez coder seul, mais n'avez-vous jamais envisagé d'intégrer ou de constituer une petite équipe ? Pourquoi ?

Le problème pour intégrer une équipe, c'est de la trouver ! La plupart des projets amateurs que je vois en présentation ont très souvent des objectifs irréalistes. Par exemple, un MMORPG. Moi je fais partie des gens qui pensent qu'un MMORPG amateur, ça n'existe pas (voir : [Lien 96](#)). Et quand en plus c'est lancé par de purs débutants, le plantage est garanti.

Même si ça me semble compliqué, je ne suis pas opposé à l'idée d'intégrer une équipe. Pour peu qu'on arrive à s'entendre, bien sûr !

### Quels conseils pourriez-vous donner aux autres ? Quelles difficultés principales avez-vous rencontrées ?

Eh bien Dam's a encore une fois tout dit ici : [Lien 97](#)

En résumé, ne pas se lancer dans un projet trop ambitieux. Par exemple, « tiens, je vais faire un petit jeu de plateformes... » Le jeu de plateformes en 2D, c'est ce qu'il y a de plus compliqué avec le RPG, donc ce n'est sûrement pas par là qu'il faut commencer. Pour un jeu de plateformes, il faut avoir un moteur de sprites, savoir gérer des animations, un scroll (tuiles + différentiel), des monstres (apparition/disparition en fonction du scroll), des tirs, etc. Et il faut aussi les outils qui vont avec, par exemple un éditeur de map. Pour info, le Mini Slug m'a pris quelque chose comme 18 mois, et ça en sachant à peu près exactement où je mettais les pieds pour avoir déjà codé pas mal de jeux de plateformes quand je travaillais dans le JV. Certes, les specs ont évolué en cours de route et moi j'aime bien tout coder moi-même, mais c'est un gros travail.

Honnêtement, plusieurs projets simples menés à terme seront bien plus bénéfiques que de se lancer dans une grosse usine à gaz interminable.

### Sur quel projet travaillez-vous actuellement ?

Hmmm... J'ai quelques projets en cours, mais je manque de plus en plus de temps. Donc je préfère ne rien dire...



Retrouvez l'article d'Alexandre Laurent et Neckara en ligne : [Lien 98](#)

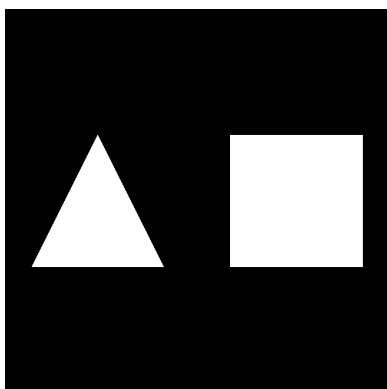
## WebGL leçon 1 : un triangle et un carré

Dans ce tutoriel, vous apprendrez à dessiner un triangle et un carré dans une page Web.

### 1. Introduction

Bienvenue dans mon premier tutoriel WebGL ! Cette première leçon repose sur le second tutoriel OpenGL de NeHe ([Lien 99](#)), qui est une voie populaire dans l'apprentissage de la 3D pour le développement de jeux. Il montre comment dessiner un triangle et un carré dans une page Web. Peut-être que cela n'est pas très passionnant en soi, mais, c'est une excellente introduction des fondements de WebGL. Si vous comprenez comment cela fonctionne, le reste devrait être simple...

Voici ce que le résultat sera lorsque vous exécuterez cette leçon dans un navigateur qui supporte WebGL :



Cliquez [ici](#) et vous verrez la version WebGL en ligne si votre navigateur le supporte : [Lien 100](#) ; cliquez [ici](#) pour en avoir un, s'il ne supporte pas WebGL : [Lien 101](#). Apprenez comment cela fonctionne ci-dessous...

Ces leçons ciblent des personnes ayant une connaissance raisonnable de la programmation, mais aucune expérience dans la 3D. Le but est de vous former, en vous donnant une explication de ce qui se passe dans le code, afin que vous puissiez produire vos propres pages Web 3D aussi vite que possible. J'écris ces articles tout en apprenant WebGL moi-même, donc il se peut (et cela est même certain !) qu'il y ait des erreurs. Utilisez-les, tout en sachant les risques : [Lien 102](#). Par contre, je corrige les bogues et les erreurs de conception lorsque je les connais, donc si vous voyez quelque chose de cassé, dites-le-moi dans les commentaires.

Il y a deux façons de récupérer le code de cet exemple : soit en affichant le code « voir source » lorsque vous regardez à la page de démonstration, soit en utilisant GitHub : vous pouvez le cloner (ainsi que les leçons suivantes) à partir du dépôt : [Lien 103](#). Dans tous les cas, dès que vous avez le code, chargez-le dans votre éditeur de texte préféré et lisez. Au premier coup d'œil, c'est plutôt intimidant, même si vous avez une connaissance innée de, disons, OpenGL. Soit, au début nous définissons un couple de « *shaders* », qui sont généralement vus comme relativement avancés... mais ne désespérez pas, c'est plus simple que cela en a l'air.

### 2. Leçon

Comme beaucoup de programmes, cette page WebGL commence par définir un ensemble de fonctions bas niveau qui sont utilisées dans le code haut niveau ci-dessous. Afin de l'expliquer, je vais commencer par la fin et remonter le tout, donc si vous suivez avec le code, sautez directement à la fin.

Vous allez voir le code HTML suivant :

```
<body onload="webGLStart();" >
  <a href="http://learningwebgl.com/blog/?p=28">&lt;&lt; Back to Lesson 1</a><br />

  <canvas id="lesson01-canvas" style="border:none;" width="500" height="500"></canvas>

  <br/>
  <a href="http://learningwebgl.com/blog/?p=28">&lt;&lt; Back to Lesson 1</a><br />
</body>
```

C'est le code complet du corps de la page : tout le reste est en JavaScript (bien que si vous visualisiez le code avec « voir source », vous pourriez voir un peu de bazar pour l'analyse de mon site Web, que vous pouvez ignorer). Évidemment nous pouvons ajouter plus de code HTML entre les balises <body> et construire notre image WebGL dans une page Web normale, mais pour cette simple démonstration nous n'avons que les liens pour revenir sur mon blog et la balise <canvas>, qui est l'emplacement de l'application 3D. Les canvas sont une nouveauté pour HTML5 : ils apportent la possibilité d'afficher des éléments dessinés par le JavaScript dans les pages Web que ce soit en 2D et (avec WebGL) en 3D. Dans cette balise, nous ne spécifions rien de plus que les propriétés de la mise en page du canvas et nous laissons tout le code d'initialisation WebGL à la fonction JavaScript webGLStart, qui, comme vous pouvez le voir, sera appelée une fois la page chargée. Défilons vers le haut pour atteindre la fonction et voyons voir à quoi elle ressemble :

```
function webGLStart()
{
  var canvas =
document.getElementById("lesson01-canvas");
  initGL(canvas);
  initShaders();
  initBuffers();

  gl.clearColor(0.0, 0.0, 0.0, 1.0);
  gl.enable(gl.DEPTH_TEST);

  drawScene();
}
```

Elle appelle les fonctions pour initialiser WebGL et les « *shaders* » que j'ai mentionnés plus tôt, en passant l'élément canvas sur lequel nous voulons dessiner nos

objets 3D à la première fonction. Ensuite, l'initialisation des tampons se fait avec la fonction `initBuffers`. Les tampons sont des choses contenant les détails du triangle et du carré que nous allons dessiner. Nous allons parler d'eux plus en détail dans un moment. Ensuite, la fonction effectue quelques initialisations basiques de WebGL, indiquant que lors de l'effacement du canvas nous devons le rendre noir et que nous devons effectuer le test de profondeur (afin que les objets dessinés derrière d'autres soient cachés par ceux devant eux). Ces étapes sont implémentées en appelant les méthodes de l'objet `gl`. Nous allons voir comment il est initialisé plus tard. Finalement, la fonction appelle `drawScene`. Celle-ci (comme vous pouvez vous y attendre) dessine le triangle et le carré en utilisant les tampons.

Nous allons revenir sur `initGL` et `initShaders` plus tard, car elles sont importantes pour la compréhension du fonctionnement de la page, mais avant, regardons les fonctions `initBuffers` et `drawScene`.

Découvrons pas à pas `initBuffers` :

```
var triangleVertexPositionBuffer;
var squareVertexPositionBuffer;
```

Nous déclarons deux variables pour contenir les tampons. (Dans une page WebGL réelle, vous n'aurez pas de variable globale pour chaque objet dans la scène, mais nous le faisons ici pour garder les choses simples, comme nous ne faisons que commencer.)

Ensuite :

```
function initBuffers() {
    triangleVertexPositionBuffer =
gl.createBuffer();
```

Nous créons un tampon pour la position des vertex du triangle. Les vertex sont des points dans l'espace 3D qui définissent la forme de ce que nous dessinons. Pour notre triangle, nous allons avoir trois vertex (qui seront initialisés dans une minute). Le tampon est un espace mémoire sur la carte graphique. En envoyant les positions des vertex sur la carte à partir de notre code d'initialisation, nous allons pouvoir dessiner la scène, principalement en disant à WebGL de « dessiner les choses que nous lui avons indiquées avant ». Cela rend notre code très efficace, spécialement lorsque nous animons la scène et que nous voulons dessiner l'objet une dizaine de fois chaque seconde afin de le déplacer. Bien sûr, lorsque ce ne sont que trois positions de vertex comme dans ce cas, le coût d'envoi des données sur la carte graphique n'est pas énorme, mais si nous jouons avec d'immenses modèles contenant des dizaines de centaines de vertex, il peut y avoir un réel avantage de faire de cette façon. Ensuite :

```
gl.bindBuffer(gl.ARRAY_BUFFER,
triangleVertexPositionBuffer);
```

Cette ligne indique à WebGL que les opérations suivantes sur les tampons doivent utiliser celui spécifié. Il y a toujours ce concept de « tableau tampon actif » et les fonctions agissent sur celui-ci plutôt que de vous laisser spécifier le tableau tampon avec lequel vous voulez travailler. Étrange, mais je suis certain que les raisons sont liées aux performances...

```
var vertices = [
    0.0, 1.0, 0.0,
    -1.0, -1.0, 0.0,
    1.0, -1.0, 0.0
];
```

Ensuite, nous définissons la position de nos vertex dans une liste JavaScript. Vous pouvez voir qu'ils forment un triangle isocèle centré en (0, 0, 0).

```
gl.bufferData(gl.ARRAY_BUFFER, new
Float32Array(vertices), gl.STATIC_DRAW);
```

Maintenant, nous créons un objet `Float32Array` basé sur notre liste JavaScript et nous indiquons à WebGL de l'utiliser pour remplir le tampon actuellement actif, qui, bien entendu, est `triangleVertexPositionBuffer`. Nous allons parler des `Float32Array` dans une leçon future, mais pour le moment tout ce que nous avons besoin de savoir est que c'est une technique pour transformer une liste JavaScript en une chose que l'on peut passer à WebGL afin de remplir ses tampons.

```
triangleVertexPositionBuffer.itemSize = 3;
triangleVertexPositionBuffer.numItems = 3;
```

La dernière chose que nous devons effectuer avec le tampon est de définir deux nouvelles propriétés. Elles ne proviennent pas de WebGL, mais elles seront très utiles plus tard. Un bon point (certains diraient un mauvais point) du JavaScript est qu'un objet n'a pas besoin de supporter explicitement une propriété spécifique pour que vous puissiez la définir. Donc même si les objets tampons n'avaient pas les propriétés `itemSize` et `numItems`, maintenant ils les ont. Nous les utilisons pour dire que ce tampon de neuf éléments représente trois positions de vertex différentes (`numItems`), composés eux-mêmes de trois nombres (`itemSize`).

Maintenant que nous avons défini complètement le tampon pour le triangle, nous pouvons faire de même pour le carré :

```
squareVertexPositionBuffer =
gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER,
squareVertexPositionBuffer);
vertices = [
    1.0, 1.0, 0.0,
    -1.0, 1.0, 0.0,
    1.0, -1.0, 0.0,
    -1.0, -1.0, 0.0
];
gl.bufferData(gl.ARRAY_BUFFER, new
Float32Array(vertices), gl.STATIC_DRAW);
squareVertexPositionBuffer.itemSize = 3;
squareVertexPositionBuffer.numItems = 4;
}
```

Tout cela devrait être assez évident. Le carré possède quatre positions de vertex au lieu de trois et donc le tableau est plus grand et `numItems` diffère.

OK, donc nous avons ce qu'il nous faut pour envoyer les positions de vertex des deux objets à la carte graphique. Maintenant, regardons la fonction `drawScene`, qui est l'endroit où nous utilisons ces tampons pour dessiner l'image que nous allons voir. Explorons-la étape par étape :



```
function drawScene() {
    gl.viewport(0, 0, gl.viewportWidth,
gl.viewportHeight);
```

La première étape est de lui donner des indications sur la taille du canvas avec la fonction `viewport`. Nous reviendrons sur le pourquoi il est important de le faire dans une prochaine leçon. Vous devez juste savoir que cette fonction doit être appelée avec la taille du canvas avant de commencer à dessiner. Ensuite, nous effaçons le canvas afin de le préparer pour notre dessin :

```
gl.clear(gl.COLOR_BUFFER_BIT |
gl.DEPTH_BUFFER_BIT);
```

... puis :

```
mat4.perspective(45, gl.viewportWidth /
gl.viewportHeight, 0.1, 100.0, pMatrix);
```

Ici, nous initialisons la perspective avec celle par laquelle nous voulons voir la scène. Par défaut, WebGL dessinera les choses proches avec la même taille que celles qui sont lointaines (dans un style 3D connu sous le nom de « *projection orthogonale* »). Afin de rendre les objets lointains plus petits, nous devons indiquer la perspective que nous voulons utiliser. Pour cette scène, nous définissons notre champ de vision vertical à 45 °, nous indiquons aussi le rapport largeur sur hauteur de notre canvas et que nous ne voulons pas voir les choses plus proches que 0,1 et plus loin que 100 unités.

Comme vous pouvez le constater, la fonction `perspective` est fournie par le module `mat4` et met en scène une intrigante variable nommée `pMatrix`. Plus d'informations viendront par la suite. J'espère pour le moment que sans connaître les détails, son utilisation est claire.

Maintenant que nous avons notre perspective définie, nous pouvons continuer à dessiner nos objets :

```
mat4.identity(mvMatrix);
```

La première étape consiste à se « déplacer » vers le centre de la scène 3D. Dans OpenGL, lorsque vous dessinez la scène, vous lui indiquez de dessiner chaque objet à un endroit « spécifique » avec une rotation « spécifique ». Donc, par exemple, vous dites « déplace-toi de 20 unités en avant, tourne de 32 degrés puis dessine le robot », la dernière partie étant un ensemble complexe d'instructions « déplace-toi de tant, tourne un peu, dessine ceci » en elle-même. C'est utile, car vous pouvez encapsuler le code « dessine un robot » dans une fonction puis déplacer facilement le robot en changeant le code de déplacement/rotation avant d'appeler la fonction.

La position et la rotation actuelles sont toutes les deux déterminées par la matrice. Comme vous l'avez sûrement appris à l'école, les matrices peuvent décrire les translations (déplacement d'un endroit à un autre), les rotations et les autres transformations géométriques. Pour des raisons que je n'approfondirai pas maintenant, vous pouvez utiliser une seule matrice 4x4 (et non 3x3) pour représenter n'importe quel nombre de transformations dans un espace 3D. Vous démarrez avec la matrice d'identité, une matrice qui représente une transformation ne faisant rien, puis vous la multipliez par la matrice qui décrit votre première transformation puis par la matrice de votre

seconde transformation et ainsi de suite. La matrice résultante représente toutes vos transformations en une seule. La matrice que nous utilisons pour représenter le déplacement et la rotation actuelles est appelée la matrice modèle-vue (*model-view*) et vous avez sûrement deviné que notre variable `mvMatrix` contient notre matrice modèle-vue et que la fonction `mat4.identity` que nous venons d'appeler définit la matrice comme matrice d'identité prête à recevoir nos translations et rotations. En d'autres mots, la matrice nous déplace vers un point à partir duquel nous commençons à dessiner le monde 3D.

Les lecteurs vigilants auront remarqué qu'au début de cette discussion sur les matrices j'ai dit « Dans OpenGL », et non pas « Dans WebGL ». Cela est dû au fait que WebGL n'intègre pas ces choses à la bibliothèque graphique. À la place, nous utilisons une bibliothèque de matrices externe, l'excellente `glMatrix` ([Lien 104](#)) de Brandon Jones avec d'élégantes astuces WebGL afin d'obtenir le même effet. D'autres informations sur cette élégance plus tard.

Soit, continuons avec le code qui dessine le triangle sur la partie gauche de notre canvas.

```
mat4.translate(mvMatrix, [-1.5, 0.0, -7.0]);
```

Après s'être placé au centre de la scène 3D en définissant la matrice `mvMatrix` à l'identité, nous commençons le triangle en nous déplaçant de 1,5 unité sur la gauche (qui est dans le sens négatif sur l'axe des X) et de sept unités dans la scène (qui est, en s'éloignant de la caméra, le sens négatif sur l'axe des Z). (`mat4.translate`, comme vous pouvez le deviner, signifie « multiplier la matrice donnée par une matrice de translation construite à partir des paramètres suivants ».)

La prochaine étape est de dessiner quelque chose !

```
gl.bindBuffer(gl.ARRAY_BUFFER,
triangleVertexPositionBuffer);
```

```
gl.vertexAttribPointer(shaderProgram.vertexPositionAttribute,
triangleVertexPositionBuffer.itemSize, gl.FLOAT,
false, 0, 0);
```

Donc, vous vous rappelez que pour utiliser un de nos tampons, vous devez appeler `gl.bindBuffer` pour indiquer le tampon sur lequel vous voulez intervenir et ensuite appeler le code agissant sur celui-ci. Ici, nous sélectionnons notre `triangleVertexPositionBuffer` puis nous indiquons à WebGL que les valeurs à l'intérieur doivent être utilisées comme positions de vertex. J'expliquerai un peu plus le fonctionnement après. Pour le moment, vous pouvez voir que nous utilisons la propriété `itemSize` que nous avons définie sur le buffer pour dire à WebGL que chaque élément dans le tampon contenait trois nombres.

Ensuite, nous avons :

```
setMatrixUniforms();
```

Cela indique à WebGL de prendre en compte notre matrice modèle-vue actuelle (et aussi la matrice de projection que nous verrons plus tard). Cela est nécessaire, car les matrices ne sont pas intégrées à WebGL. La façon de procéder est que vous pouvez faire tout ce que vous voulez en changeant la variable `mvMatrix` mais cela ne se produit que dans l'espace privé de JavaScript. `setMatrixUniforms` est une fonction définie plus haut dans ce fichier pour

envoyer la matrice à la carte graphique.

Une fois que cela est fait, WebGL possède un tableau de nombres qui doivent être traités comme des positions de vertex et il connaît nos matrices. La prochaine étape est de lui dire ce qu'il doit en faire :

```
gl.drawArrays(gl.TRIANGLES, 0,
triangleVertexPositionBuffer.numItems);
```

Ou, dit autrement : « dessine les tableaux de vertex que je t'ai donnés précédemment comme triangles en commençant par l'élément 0 du tableau jusqu'à l'élément numItems ».

Une fois cela fait, WebGL a dessiné notre triangle. La prochaine étape est de dessiner notre carré :

```
mat4.translate(mvMatrix, [3.0, 0.0, 0.0]);
```

Nous commençons par déplacer notre matrice de modèle-vue de trois unités vers la droite. Rappelez-vous, nous sommes actuellement 1,5 unité sur la gauche et sept en avant. Ensuite :

```
gl.bindBuffer(gl.ARRAY_BUFFER,
squareVertexPositionBuffer);

gl.vertexAttribPointer(shaderProgram.vertexPositionAttribute, squareVertexPositionBuffer.itemSize,
gl.FLOAT, false, 0, 0);
```

Donc, nous indiquons à WebGL d'utiliser notre tampon pour le carré comme positions de vertex...

```
setMatrixUniforms();
```

... nous envoyons les matrices de modèle-vue et de projection une nouvelle fois (afin de prendre en compte le dernier mvTranslate), signifiant que nous pouvons finalement :

```
gl.drawArrays(gl.TRIANGLE_STRIP, 0,
squareVertexPositionBuffer.numItems);
```

dessiner les points. Vous pouvez vous demander ce qu'est un « *triangle strip* » ? Eh bien, c'est une suite de triangles :-). Plus précisément, c'est une suite de triangles où les trois premiers vertex que vous donnez forment le premier triangle, puis les deux derniers de celui-ci plus le vertex suivant forment le second triangle et ainsi de suite. Dans ce cas, c'est une méthode simple et sale pour construire un carré. Dans des cas plus complexes, cela peut être une technique très utile pour construire des surfaces complexes en termes du nombre de triangles.

Bref, une fois cela fait, nous avons fini notre fonction drawScene.

```
}
```

Si vous êtes arrivé aussi loin, vous êtes définitivement prêt à expérimenter. Copiez le code dans un fichier local que ce soit à partir de GitHub ([Lien 105](#)) ou du fichier de démonstration. Pour ce dernier, vous avez besoin de index.html ([Lien 106](#)) et glMatrix-0.9.5.min.js ([Lien 107](#)). Exécutez en local pour être certain que cela fonctionne, puis essayez de changer quelques positions de vertex ci-dessus. En particulier, la scène actuelle est plutôt plate.

Essayez de changer les valeurs sur l'axe des Z pour le carré à 2 ou -3 et voyez s'il devient plus grand ou plus petit selon son déplacement en avant ou en arrière. Ou essayez de modifier juste un ou deux vertex et observez la déformation suivant la perspective. Amusez-vous et ne vous occupez pas de moi, je vais attendre.

...

OK, vous êtes revenu. Regardons les fonctions de support qui ont rendu possible l'exécution du code que nous avons vu. Comme je l'ai dit avant, si vous vous contentez d'ignorer les détails et que vous copiez juste les fonctions d'aide situées au-dessus de initBuffers, vous pouvez probablement partir avec et construire des pages WebGL intéressantes (bien qu'uniquement en noir et blanc, les couleurs étant le sujet de la prochaine leçon : [Lien 108](#)). Mais aucun de ces détails n'est difficile à comprendre et en apprenant comment cela fonctionne vous écrirez sûrement du meilleur code WebGL par la suite.

Toujours avec moi ? Merci :-). Débarrassons-nous des fonctions les plus ennuyantes en premier. La première est appelée par webGLStart et c'est initGL. Elle est proche du haut de la page et voici une copie :

```
var gl;
function initGL(canvas) {
  try {
    gl = canvas.getContext("experimental-
webgl");
    gl.viewportWidth = canvas.width;
    gl.viewportHeight = canvas.height;
  } catch(e) {
  }
  if (!gl) {
    alert("Could not initialise WebGL, sorry :-
");
  }
}
```

C'est très simple. Comme vous pouvez le remarquer, les fonctions initBuffers et drawScene utilisent fréquemment un objet appelé gl, qui peut être identifié comme une sorte de « cœur » de WebGL. Cette fonction récupère ce « cœur », appelé contexte WebGL, et le récupère en le demandant au canvas passé en paramètre grâce à un nom standard de contexte. (Comme vous pouvez le deviner, un jour le nom du contexte changera de « experimental-webgl » pour devenir « webgl »). Je mettrai à jour cette leçon et mon blog lorsque cela arrivera. Abonnez-vous à mon flux RSS ([Lien 109](#)) si vous souhaitez être prévenu et bien sûr, si vous souhaitez avoir des nouvelles sur WebGL toutes les semaines.) Une fois que nous avons le contexte, nous utilisons encore les avantages du JavaScript nous permettant d'ajouter n'importe quelle propriété sur n'importe quel objet afin de sauvegarder la largeur et hauteur du canvas associé. Ainsi, nous pouvons utiliser ces propriétés dans le code pour définir le viewport et la perspective ([Lien 110](#)) au début de la fonction startScene. Une fois que cela est fait, notre contexte GL est prêt.

Après avoir appelé initGL, webGLStart appelle la fonction initShaders. Celle-ci, bien entendu, initialise les shaders (doh ;-)). Nous reviendrons sur celle-ci plus tard, car nous devons d'abord regarder notre matrice modèle-vue et notre matrice de projection que j'ai mentionnées plus tôt. Voici le code :

```
var mvMatrix = mat4.create();
```



```
var pMatrix = mat4.create();
```

Donc, nous définissons une variable appelée `mvMatrix` pour contenir la matrice de modèle-vue et une autre appelée `pMatrix` pour la matrice de projection et nous les initialisons à une valeur vide (tous zéros) pour commencer. Il est maintenant nécessaire d'en dire un peu plus sur la matrice de projection. Comme vous vous en souvenez, nous appliquons la fonction `glMatrix mat4.perspective` à cette variable pour définir notre perspective, tout au début de la fonction `drawScene`. Cela est dû au fait que WebGL ne supporte pas directement les perspectives, tout comme il ne supporte pas directement une matrice modèle-vue. Mais, tout comme le déplacement des objets et leur rotation qui sont encapsulés dans la matrice modèle-vue, le processus pour rendre proportionnellement les objets lointains plus petits que les objets proches est une des choses dans lesquelles les matrices excellent. Et, comme vous devez l'avoir deviné, la matrice de projection est celle qui le fait. La fonction `mat4.perspective`, avec son ratio et son champ de vision, remplit la matrice avec les valeurs qui donnent le genre de perspective que nous voulons.

Bien, maintenant nous avons tout vu sauf la fonction `setMatrixUniforms`, qui comme je l'ai dit précédemment, transfère les matrices de modèle-vue et projection de JavaScript à WebGL ainsi que l'effrayante gestion des shaders. Ceux-ci sont interconnectés, donc commençons par une introduction.

Vous pouvez vous demander ce qu'est un shader ? Bien, à un certain moment de l'histoire des graphismes 3D ils étaient possiblement ce qu'ils sous-entendaient : un morceau de code indiquant au système comment ombrer ou colorer des morceaux de la scène avant son affichage. Par contre, avec le temps, leurs objectifs se sont étendus à tel point qu'il serait mieux de les définir comme un morceau de code qui peut faire tout ce que nous voulons sur des morceaux de la scène avant que celle-ci ne soit dessinée. Et cela est très pratique, car (a) ils sont exécutés sur la carte graphique, donc ils font ce qu'ils doivent faire très rapidement et (b) le genre de transformations qu'ils peuvent faire est très pratique pour les petits exemples comme ceux-ci.

La raison pour laquelle nous parlons des shaders dans un exemple simple de WebGL (ils sont de niveau intermédiaire dans les tutoriels OpenGL) est que nous les utilisons pour récupérer le système WebGL, fonctionnant heureusement sur la carte graphique. Les shaders permettent d'appliquer notre matrice de modèle-vue et notre matrice de projection de notre scène sans avoir à déplacer tous les points et tous les vertex en JavaScript, ce qui serait (relativement) lent. C'est très pratique et évite un surcoût.

Donc, voici comment ils sont définis. Comme vous vous en souvenez, `webGLStart` appelle `initShaders`, donc regardons-la, étape par étape.

```
var shaderProgram;
function initShaders() {
    var fragmentShader = getShader(gl, "shader-
fs");
    var vertexShader = getShader(gl, "shader-
vs");

    shaderProgram = gl.createProgram();
    gl.attachShader(shaderProgram, vertexShader);
    gl.attachShader(shaderProgram,
fragmentShader);
```

```
gl.linkProgram(shaderProgram);

    if (!gl.getProgramParameter(shaderProgram,
gl.LINK_STATUS)) {
        alert("Could not initialise shaders");
    }

    gl.useProgram(shaderProgram);
```

Comme vous pouvez le voir, elle utilise une fonction appelée `getShader` pour récupérer deux choses, un « *fragment shader* » et un « *vertex shader* » pour les attacher ensuite à un truc WebGL appelé « *programme* ». Un programme est un morceau de code qui vit du côté WebGL du système. Vous pouvez le considérer comme une façon de spécifier quelque chose qui peut être exécuté sur la carte graphique. Comme vous pouvez vous y attendre, vous pouvez l'associer à un nombre de shaders contenant chacun un morceau de code de ce programme. Précisément, chaque programme peut contenir un « *fragment* » et un « *vertex shader* ». Nous allons les détailler rapidement.

```
    shaderProgram.vertexPositionAttribute =
gl.getAttribLocation(shaderProgram,
"aVertexPosition");

gl.enableVertexAttribArray(shaderProgram.vertexPo
sitionAttribute);
```

Une fois que la fonction a initialisé et attaché les « *shaders* », elle récupère une référence sur un « *attribut* », qui sera contenue dans un nouveau champ de l'objet du programme appelé `vertexPositionAttribute`. Une fois encore, nous utilisons les avantages de JavaScript pour définir n'importe quel champ de n'importe quel objet. Les objets programmes n'ont pas de champ `vertexPositionAttribute` par défaut, mais il nous est pratique de garder les deux valeurs ensemble, donc nous mettons l'attribut dans le nouveau champ du programme. Donc, à quoi sert `vertexPositionAttribute` ? Comme vous pouvez vous en souvenir, nous l'utilisons dans `drawScene`. Si vous regardez le code qu'il définit ([Lien 111](#)), vous allez voir que la chose que nous avons associée au tampon est cet attribut. Vous allez voir ce que cela signifie dans un moment. Pour l'instant, notez simplement que nous utilisons aussi `gl.enableVertexAttribArray` pour indiquer à WebGL que nous voulons fournir des valeurs pour l'attribut par le biais d'un tableau.

```
    shaderProgram.pMatrixUniform =
gl.getUniformLocation(shaderProgram, "uPMatrix");
    shaderProgram.mvMatrixUniform =
gl.getUniformLocation(shaderProgram,
"uMVMMatrix");
}
```

La dernière chose qu'effectue `initShaders` est de récupérer deux autres valeurs du programme : l'emplacement des variables appelées « *variables uniformes* ». Nous allons bientôt les rencontrer. Pour le moment, vous devez juste retenir qu'elles sont comme l'attribut, nous les stockons dans l'objet du programme par facilité.

Maintenant, jetons un coup d'œil à `getShader` :

```
function getShader(gl, id) {
    var shaderScript =
```

```

document.getElementById(id);
    if (!shaderScript) {
        return null;
    }

    var str = "";
    var k = shaderScript.firstChild;
    while (k) {
        if (k.nodeType == 3)
            str += k.textContent;
        k = k.nextSibling;
    }

    var shader;
    if (shaderScript.type == "x-shader/x-
fragment") {
        shader =
gl.createShader(gl.FRAGMENT_SHADER);
    } else if (shaderScript.type == "x-
shader/x-vertex") {
        shader =
gl.createShader(gl.VERTEX_SHADER);
    } else {
        return null;
    }

    gl.shaderSource(shader, str);
    gl.compileShader(shader);

    if (!gl.getShaderParameter(shader,
gl.COMPILE_STATUS)) {
        alert(gl.getShaderInfoLog(shader));
        return null;
    }

    return shader;
}

```

C'est une autre de ces fonctions qui est plus simple qu'il n'y paraît. Tout ce que nous faisons ici est de chercher un élément dans notre page HTML qui a un identifiant qui correspond au paramètre passé, l'extrayant du contenu de la page pour en créer un fragment ou un vertex shader selon son type (nous verrons dans une future leçon la différence entre les deux). Finalement, le shader produit est passé à WebGL afin qu'il soit compilé sous une forme exécutable pour la carte graphique. Le code gère les erreurs et c'est bon ! Bien sûr, nous pouvons simplement définir les shaders comme chaînes de caractères dans le code JavaScript et ne pas nous ennuyer avec leur extraction du HTML. Mais en faisant de la sorte, nous pouvons les rendre plus faciles à lire, car ils sont définis comme script de la page Web, comme s'ils étaient du JavaScript eux-mêmes.

Une fois la fonction analysée, nous pouvons regarder le code des shaders :

```

<script id="shader-fs" type="x-shader/x-
fragment">
    precision mediump float;

    void main(void) {
        gl_FragColor = vec4(1.0, 1.0, 1.0, 1.0);
    }
</script>

<script id="shader-vs" type="x-shader/x-vertex">
    attribute vec3 aVertexPosition;

```

```

uniform mat4 uMVMatrix;
uniform mat4 uPMatrix;

void main(void) {
    gl_Position = uPMatrix * uMVMatrix *
vec4(aVertexPosition, 1.0);
}
</script>

```

La première chose qu'il faut se rappeler est qu'ils **ne** sont **pas** écrits en JavaScript, même si l'ancêtre du langage est très proche. En vrai, ils sont écrits dans un langage spécifique au shader : le GLSL, partageant beaucoup avec le C (comme le fait JavaScript, bien entendu).

Le premier est le fragment shader et ne fait presque rien. Il contient un morceau de code obligatoire pour indiquer à la carte graphique la précision que nous souhaitons pour les nombres à virgule flottante (la précision « medium » est bonne car elle est supportée par tous les périphériques WebGL. La précision « highp » pour la haute précision ne fonctionne pas sur les mobiles), puis indique que tout ce qui doit être dessiné doit être blanc. (La prochaine leçon expliquera comment colorer les objets : [Lien 112.](#))

Le second shader est un peu plus intéressant. C'est un vertex shader, qui, je vous le rappelle, est un morceau de code pour la carte graphique qui peut faire à peu près tout ce qu'il veut avec le vertex. Associé à lui, il possède deux variables uniformes appelées uMVMatrix et uPMatrix. Les variables uniformes sont utiles, car elles peuvent être accessibles depuis l'extérieur du shader, évidemment depuis l'extérieur du programme et plus précisément à partir des emplacements extraits avec la fonction initShaders et définies avec les valeurs des matrices de modèle-vue et de projection en utilisant le code ci-dessous. Vous pouvez comparer le programme shader à un objet (dans le sens orienté objet) et les variables uniformes comme des membres.

Maintenant, le shader est appelé pour tous les vertex et les vertex sont passés dans le code du shader en utilisant la variable aVertexPosition, grâce à l'utilisation de vertexPositionAttribute de la fonction drawScene lorsque nous avons associé l'attribut au tampon. Le petit morceau de code de la fonction main du shader ne fait que multiplier la position du vertex par les matrices de modèle-vue et de projection et envoie le résultat comme position finale du vertex.

Donc, webGLStart a appelé initShaders, qui a utilisé getShader pour charger le fragment et le vertex shader à partir des scripts de la page Web, afin qu'ils puissent être compilés et passés à WebGL pour être utilisés plus tard lors du rendu de notre scène 3D.

Après ça, le seul morceau de code non expliqué est setMatrixUniforms, qui est facile à comprendre une fois que vous connaissez tout ce qui est au-dessus. :-)

```

function setMatrixUniforms() {

gl.uniformMatrix4fv(shaderProgram.pMatrixUniform,
false, pMatrix);

gl.uniformMatrix4fv(shaderProgram.mvMatrixUniform,
false, mvMatrix);
}

```

Donc, en utilisant les références aux variables uniformes

qui représentent notre matrice de projection et notre matrice modèle-vue récupérées de la fonction `initShaders`, nous envoyons à WebGL les valeurs de nos matrices JavaScript.

Ouf ! C'était immense pour une première leçon, mais heureusement que maintenant vous (et je) comprenez mieux le fonctionnement de la base nécessaire pour construire quelque chose de plus intéressant, coloré, animé

et avec des modèles WebGL à trois dimensions. Pour découvrir tout cela, lisez la seconde leçon : [Lien 113](#).

*Retrouvez l'article de Giles Thomas traduit par Alexandre Laurent en ligne : [Lien 114](#)*

*Cet article s'inscrit dans une série en cours de traduction. Vous pouvez retrouver les autres chapitres dans ce sommaire : [Lien 115](#)*

# Liens

- Lien 01 : <http://fr.wikipedia.org/wiki/Kelvin>  
Lien 02 : <http://www.httr.ups-tlse.fr/pedagogie/cours/fibre/fotheori.htm>  
Lien 03 : <http://www.httr.ups-tlse.fr/pedagogie/index.html>  
Lien 04 : <http://reseau.developpez.com/tutoriels/caleca/la-fibre-optique/>  
Lien 05 : <http://mac.developpez.com/livres/>  
Lien 06 : <http://mac.developpez.com/cours/actiondossierapplescript/>  
Lien 07 : [http://symfony.com/doc/master/book/from\\_flat\\_php\\_to\\_symfony2.html](http://symfony.com/doc/master/book/from_flat_php_to_symfony2.html)  
Lien 08 : <https://github.com/bpesquet/MonBlog/tree/sans-mvc>  
Lien 09 : <http://php.net/manual/fr/function.echo.php>  
Lien 10 : <http://php.net/manual/fr/control-structures.alternative-syntax.php>  
Lien 11 : <http://php.net/manual/fr/book.pdo.php>  
Lien 12 : <https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-1-basic-coding-standard.md>  
Lien 13 : <http://php.net/manual/fr/language.basic-syntax.phptags.php>  
Lien 14 : <http://php.net/manual/fr/function.ob-start.php>  
Lien 15 : <http://php.net/manual/fr/function.ob-get-clean.php>  
Lien 16 : <http://bpesquet.developpez.com/tutoriels/php/evoluer-architecture-mvc/>  
Lien 17 : <http://pckult.developpez.com/tutoriels/javascript/frameworks/jquery/introduction/>  
Lien 18 : <http://api.jquery.com/>  
Lien 19 : <http://dmourouval.developpez.com/tutoriels/jquery/selecteurs-personnalises/>  
Lien 20 : [http://fr.wikipedia.org/wiki/Ne\\_vous\\_répétez\\_pas](http://fr.wikipedia.org/wiki/Ne_vous_répétez_pas)  
Lien 21 : <http://zeroturnaround.com/software/jrebel/download/>  
Lien 22 : <http://zeroturnaround.com/software/jrebel/maven/>  
Lien 23 : <https://github.com/hlassiege/maven-tomcat-jetty/tree/master/1.developpez-webapp>  
Lien 24 : <http://logback.qos.ch/>  
Lien 25 : [http://fr.wikipedia.org/wiki/Representational\\_State\\_Transfer](http://fr.wikipedia.org/wiki/Representational_State_Transfer)  
Lien 26 : <https://github.com/hlassiege/maven-tomcat-jetty/tree/master/2.developpez-webapp-jaxrs>  
Lien 27 : <https://github.com/hlassiege/maven-tomcat-jetty/blob/master/2.developpez-webapp-jaxrs/src/test/resources/logback.xml>  
Lien 28 : <https://github.com/hlassiege/maven-tomcat-jetty/tree/master/3.developpez-webapp-jndi>  
Lien 29 : <https://github.com/hlassiege/maven-tomcat-jetty/blob/master/3.developpez-webapp-jndi/src/test/resources/context.xml>  
Lien 30 : <http://mojo.codehaus.org/sql-maven-plugin/>  
Lien 31 : <https://github.com/hlassiege/maven-tomcat-jetty/blob/master/3.developpez-webapp-jndi/src/test/resources/jetty-ds.xml>  
Lien 32 : <http://maven.apache.org/surefire/maven-failsafe-plugin/>  
Lien 33 : <http://en.wikipedia.org/wiki/Fail-fast>  
Lien 34 : <https://github.com/hlassiege/maven-tomcat-jetty>  
Lien 35 : <https://github.com/hlassiege/art-maven-tomcat-jetty>  
Lien 36 : <http://blog.jetoile.fr/>  
Lien 37 : <http://aldian.developpez.com/cours/le-debugage-en-java-javace/>  
Lien 38 : <http://hugo.developpez.com/tutoriels/java/developpement-web-avec-maven-tomcat-et-jetty/>  
Lien 39 : [http://www.youtube.com/watch?v=omuW5M1\\_s2E](http://www.youtube.com/watch?v=omuW5M1_s2E)  
Lien 40 : <http://www.developpez.net/forums/d1342402/java/edi-outils-pour-java/autres-edi/intellij/intellij-idea-sort-en-version-121/>  
Lien 41 : <http://www.developpez.net/forums/d1208364/java/general-java/oracle-devoile-roadmap-jdk-8/#post7264215>  
Lien 42 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/Remote.html>  
Lien 43 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/server/UnicastRemoteObject.html>  
Lien 44 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/Naming.html>  
Lien 45 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/registry/Registry.html>  
Lien 46 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/registry/LocateRegistry.html>  
Lien 47 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/server/UnicastRemoteObject.html>  
Lien 48 : <http://docs.oracle.com/javase/7/docs/api/javax/rmi/PortableRemoteObject.html>  
Lien 49 : <http://docs.oracle.com/javase/7/docs/api/java/rmi/Naming.html>  
Lien 50 : <http://docs.oracle.com/javase/7/docs/api/javax/naming/InitialContext.html>  
Lien 51 : <http://dcabasson.developpez.com/articles/java/maven/introduction-maven2/>  
Lien 52 : <http://matthieu-lux.developpez.com/tutoriels/java/maven/>  
Lien 53 : <http://jpinnetti.developpez.com/tutoriel/java/utilisation-rmi-avec-iiop/>  
Lien 54 : <http://www.developpez.com/actu/55538/>  
Lien 55 : <http://www.developpez.com/actu/55459/>  
Lien 56 : <http://www.developpez.com/actu/55543/>  
Lien 57 : <http://www.developpez.com/actu/55496/>  
Lien 58 : <http://www.developpez.com/actu/55515/>  
Lien 59 : <http://www.developpez.com/actu/55507/>  
Lien 60 : <http://www.developpez.com/actu/55551/>  
Lien 61 : <http://www.developpez.com/actu/55599/>  
Lien 62 : <http://www.developpez.net/forums/d1344183/club-professionnels-en-informatique/actualites/pdg-google-nous-sommes-a-1-ce-qui-est-possible/>  
Lien 63 : <http://www.developpez.net/forums/d1343402/java/general-java/java-mobiles/android/google-sort-nouvel-edi-pour-android-android-studio/>  
Lien 64 : <http://www.developpez.net/forums/d1238312-6/club-professionnels-en-informatique/actualites/google-glass-en-precommande-a-1-500/#post7272669>  
Lien 65 : <http://windows.developpez.com/faq/xp/?page=baseregistre>  
Lien 66 : <http://search.cpan.org/dist/Win32-TieRegistry/>  
Lien 67 : <http://windows.developpez.com/faq/xp/?page=baseregistre#restaurer-sauvegarde-base-de-registre>  
Lien 68 : <http://djabril.developpez.com/tutoriels/perl/installation-modules/>  
Lien 69 : <http://msdn.microsoft.com/en-us/library/windows/desktop/ms724072%28v=vs.85%29.aspx>  
Lien 70 : <http://djabril.developpez.com/tutoriels/perl/win32-base-registre/fichiers/sources.zip>  
Lien 71 : <http://djabril.developpez.com/tutoriels/perl/win32-base-registre/>  
Lien 72 : <http://www.libreoffice.org/>  
Lien 73 : <http://www.documentfoundation.org/>  
Lien 74 : <http://www.libreoffice.org/download/system-requirements/>  
Lien 75 : <http://www.libreoffice.org/>

- Lien 76 : <http://vviale.developpez.com/tutoriels/openoffice-libreoffice/installation/>
- Lien 77 : <http://extensions.libreoffice.org/>
- Lien 78 : <http://vviale.developpez.com/tutoriels/openoffice-libreoffice/installation/#LIII>
- Lien 79 : <https://wiki.documentfoundation.org/FR/FAQ>
- Lien 80 : <http://www.libreoffice.org/get-help/documentation/>
- Lien 81 : <http://fr.libreoffice.org/forums/>
- Lien 82 : <http://www.libreoffice.org/international-sites/>
- Lien 83 : [https://wiki.documentfoundation.org/Local\\_Mailing\\_Lists](https://wiki.documentfoundation.org/Local_Mailing_Lists)
- Lien 84 : <http://www.libreoffice.org/get-help/accessibility/>
- Lien 85 : <http://www.libreoffice.org/get-help/installation/>
- Lien 86 : <http://openoffice-libreoffice.developpez.com/tutoriels/libreoffice/presentation/>
- Lien 87 : <http://pouet.net/groups.php?which=1587>
- Lien 88 : <http://www.clubevolution4.com/HunoPortSDL/>
- Lien 89 : <http://www.youtube.com/v/gKNvLyxHTmg>
- Lien 90 : [http://www.youtube.com/v/0bZbOH\\_1sj8](http://www.youtube.com/v/0bZbOH_1sj8)
- Lien 91 : <http://www.youtube.com/v/pLID3k3rov0>
- Lien 92 : [http://www.youtube.com/v/DvQ\\_PRw\\_45E](http://www.youtube.com/v/DvQ_PRw_45E)
- Lien 93 : [http://en.wikipedia.org/wiki/Ninety-ninety\\_rule](http://en.wikipedia.org/wiki/Ninety-ninety_rule)
- Lien 94 : <http://www.spriter-resource.com/>
- Lien 95 : <http://leonard.oxg.free.fr/>
- Lien 96 : <http://conquerirlemonde.com/blog/2010/04/25/erreur-classique-n°11-faire-un-mmo-3d-amateur/>
- Lien 97 : <http://conquerirlemonde.com/blog/2010/02/16/erreur-classique-n°10-le-plantage-post-proto/>
- Lien 98 : <http://jeux.developpez.com/interviews/c1702-developpeur-sdl/>
- Lien 99 : [http://nehe.developpez.com/?page=page\\_0#LI-2](http://nehe.developpez.com/?page=page_0#LI-2)
- Lien 100 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/1-triangle-carre-webgl/fichiers/demo1.html>
- Lien 101 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/0-debuter-webgl/>
- Lien 102 : [http://learningwebgl.com/blog/?page\\_id=2](http://learningwebgl.com/blog/?page_id=2)
- Lien 103 : <http://github.com/gpjt/webgl-lessons>
- Lien 104 : <http://code.google.com/p/glmatrix/>
- Lien 105 : <http://github.com/gpjt/webgl-lessons>
- Lien 106 : <http://learningwebgl.com/lessons/lesson01/index.html>
- Lien 107 : <http://learningwebgl.com/lessons/lesson01/glMatrix-0.9.5.min.js>
- Lien 108 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/2-ajout-de-couleurs/>
- Lien 109 : <http://learningwebgl.com/blog/?feed=rss2>
- Lien 110 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/1-triangle-carre-webgl/#viewport>
- Lien 111 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/1-triangle-carre-webgl/#attribut>
- Lien 112 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/2-ajout-de-couleurs/>
- Lien 113 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/2-ajout-de-couleurs/>
- Lien 114 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/1-triangle-carre-webgl/>
- Lien 115 : <http://jeux.developpez.com/tutoriels/OpenGL/WebGL/>