

# Developpez

## Le Mag

Édition de juin - juillet 2012.

Numéro 40.

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

C++	Page 2
Qt	Page 5
Mac	Page 13
MS Office	Page 14
JavaScript	Page 21
(X)HTML/CSS	Page 29
Java	Page 39
Eclipse	Page 46
Android	Page 49
Business Intelligence	Page 52
Liens	Page 62

### Article (X)HTML/CSS



#### Ajoutez du style à vos listes ordonnées

Ajouter des styles à des listes ordonnées a longtemps été une tâche difficile et piègeuse, je vais vous montrer comment utiliser CSS3 pour améliorer leur présentation en utilisant une approche sémantique.

par **Didier Mouronval**

Page 29

### Article Eclipse



#### Premiers pas avec Eclipse Scout

Cet article propose une présentation détaillée du framework Eclipse Scout.

par **Jérémy Bresson**

Page 46

### Éditorial

Cet été, croquez à pleines dents dans de nouveaux articles bien juteux et savourez toute la fraîcheur des news et des billets blogs de la rédaction.

Profitez-en bien !

La rédaction

### Nouveaux forums publics pour le comité de normalisation du C++

Jusqu'à maintenant, celui qui voulait faire des remarques ou propositions à propos du C++ sans pour autant être membre du comité de normalisation pouvait poster sur le forum Usenet ([Lien 01](#)) comp.std.c++ (et dans une moindre mesure comp.lang.c++.moderated : [Lien 02](#)). Bien que non officiel, ce forum était suivi par un grand nombre de membres du comité qui pouvaient ainsi relayer les avis de l'ensemble de la communauté.

Le problème est qu'Usenet n'est plus trop utilisé, et ces forums sont moribonds. Afin de renforcer les liens avec la communauté des utilisateurs du C++, le comité de normalisation a donc décidé d'ouvrir des forums Web publics reprenant ce rôle.

Deux forums sont créés :

- discussions sur le langage tel qu'il est actuellement : [Lien 03](#) ;
- propositions d'évolution du langage : [Lien 04](#).

L'objectif de ces forums est de discuter de problèmes potentiels dans la norme ou d'évolution du langage, pas de l'apprentissage du C++ ni des problèmes de son code (mais heureusement, pour ça, il y a les forums C++ de Developpez.com !)

La première annonce officielle prévue pour ces forums est pendant la conférence C++Now (ex-Boostcon) : [Lien 05](#).

Que pensez-vous de cette initiative ?

*Commentez la news de Loïc Joly en ligne : [Lien 06](#)*

### Les "Guru of the week" en français Découvrez (ou redécouvrez) les célèbres problèmes C++ de Herb Sutter

C'est une source d'information que les développeurs expérimentés connaissent bien. Guru of the Week (GotW) est un site créé et alimenté par Herb Sutter entre 1997 et 2003. Le principe est simple : une question technique est posée et les lecteurs interviennent pour répondre à la question en essayant de faire le tour de toutes les difficultés techniques qui pourraient apparaître. Une note sur 10 indique le niveau de difficulté de la question. Cette discussion aboutit à une analyse en profondeur de la problématique posée. Ces questions et réponses ont eu tellement de succès que Herb Sutter a publié plusieurs ouvrages pour regrouper et compléter ces analyses : Exceptional C++ ([Lien 07](#)), More Exceptional C++ ([Lien 08](#)) et Exceptional C++ Style ([Lien 09](#)). Pour les lecteurs francophones, seul le premier ouvrage est traduit en français.

L'équipe de rédaction de la rubrique C++ de Developpez.com ([Lien 10](#)) vous propose de découvrir (ou redécouvrir) ces articles traduits en français. En fonction des autres publications de la rubrique, nous vous proposerons ainsi un ou deux GotW par semaine.

Cette semaine, nous vous proposons le GotW numéro 31, sur les fonctions virtuelles impures : [Lien 11](#).

#### Connaissez-vous ces subtilités concernant les fonctions virtuelles pures ?

#### Que pensez-vous de cette initiative de Developpez.com de vous proposer ces articles en français ?

*Retrouvez la liste de tous les Guru of the Week sur la page d'index : [Lien 12](#).*

*Commentez la news de Guillaume Belz en ligne : [Lien 13](#)*

## Les dernier tutoriels et articles

### Guru Of the Week n° 31 : les fonctions virtuelles (im)pures

Difficulté : 7 / 10

Est-ce toujours une bonne chose de faire une fonction virtuelle pure, tout en fournissant un body ?

#### 1. Problèmes

##### 1.1. Question Junior

Qu'est-ce qu'une fonction virtuelle pure ? Donnez un exemple.

##### 1.2. Question Guru

Pourquoi déclarer une fonction virtuelle pure et aussi

écrire une définition (body) ? Donnez autant de raisons ou de situations que vous pouvez.

#### 2. Solutions

##### 2.1. Question Junior

Qu'est-ce qu'une fonction virtuelle pure ? Donnez un exemple.

Une fonction virtuelle pure est une fonction virtuelle dont vous voulez forcer l'implémentation par les classes dérivées. Si une classe comporte encore des fonctions virtuelles non redéfinies, alors c'est une classe abstraite, et vous ne pourrez pas instancier ce type.

```
class AbstractClass {
public:
    // déclare une fonction virtuelle pure :
    // cette classe est désormais abstraite
    virtual void f(int) = 0;
};

class StillAbstract : public AbstractClass {
    // impossible d'instancier f(int),
    // aussi cette classe est-elle toujours
    abstraite
};

class Concrete : public StillAbstract {
public:
    // instancie f(int),
    // aussi cette classe est concrète
    void f(int) { /*...*/ }
};

AbstractClass a; // erreur, classe abstraite
StillAbstract b; // erreur, classe abstraite
Concrete c; // ok, classe concrète
```

## 2.2. Question Guru

*Pourquoi déclarer une fonction virtuelle pure et aussi écrire une définition (body) ? Donnez autant de raisons ou de situations que vous pouvez.*

Il y a trois principales raisons pour faire cela : la première est courante, la deuxième est assez rare et la troisième est une solution de rechange utilisée de temps en temps par des programmeurs expérimentés travaillant avec des compilateurs un peu faibles. La plupart des programmeurs ne devraient jamais utiliser que la première.

### 2.2.1. Destructeur virtuel pur

Toutes les classes de base devraient avoir un destructeur virtuel (consultez votre manuel C++ favori pour y trouver les raisons).(Note de traduction : vous pouvez également consulter la FAQ « Pourquoi et quand faut-il créer un destructeur virtuel ? » : [Lien 14](#)) Si la classe doit être abstraite (vous voulez éviter son instanciation) mais qu'il s'avère qu'elle n'a aucune autre fonction virtuelle pure, voici une technique courante pour créer le destructeur virtuel pur :

```
// file b.h
class B {
public:
    /*...d'autres choses...*/

    virtual ~B() = 0; // destructeur virtuel pur
};
```

Bien sûr, tout destructeur de classe dérivée doit appeler le destructeur de classe de base, ainsi le destructeur doit rester défini (même s'il est vide) :

```
// file b.cpp
B::~B() { /* peut être vide */ }
```

Si cette définition ne devait pas être fournie, vous pourriez toujours dériver les classes de B, mais elles ne pourraient jamais être instanciées, ce qui n'est pas particulièrement utile.

### 2.2.2. Acceptation consciente forcée d'un comportement par défaut

Si une classe dérivée choisit de ne pas redéfinir une fonction virtuelle non pure, elle se contente d'hériter du comportement de base par défaut de sa version. Si vous voulez fournir un comportement par défaut, mais que vous ne voulez pas laisser les classes dérivées simplement hériter "silencieusement", vous pouvez la rendre virtuelle pure tout en lui fournissant un comportement par défaut. L'auteur de la classe dérivée pourra l'appeler s'il veut l'utiliser :

```
class B {
public:
    virtual bool f() = 0;
};

bool B::f() {
    return true; // c'est un bon comportement
                // par défaut, mais
                // il ne devrait pas être
                // utilisé en aveugle
}

class D : public B {
public:
    bool f() {
        return B::f(i); // si D veut le
        comportement par défaut,
                        // il doit le dire
    }
};
```

### 2.2.3. Solution de rechange pour les diagnostics de mauvais compilateurs

Il y a des situations dans lesquelles il se peut que vous vous retrouviez à appeler accidentellement une fonction virtuelle pure (indirectement à partir d'un constructeur ou destructeur de base ; référez-vous à votre manuel de C++ avancé favori pour avoir des exemples (Note de traduction : vous pouvez également consulter la FAQ « Puis-je appeler des fonctions virtuelles dans le constructeur (ou le destructeur) ? » : [Lien 15](#)) ). Bien sûr, un code bien écrit ne devrait pas présenter ce genre de problème, mais nul n'est parfait et ça peut toujours arriver. Malheureusement, il y a des compilateurs (c'est vrai que techniquement, c'est l'environnement de l'application qui détecte ce genre de chose. Toutefois je dirai à chaque fois "compilateur", parce que c'est généralement le compilateur qui est censé entrer dans le code qui vérifie cela pour vous quand vous lancez le programme) qui ne vous préviendront pas de ce problème. Ceux qui ne le font pas peuvent vous donner de mauvais messages d'erreur qui prennent un temps infini à comprendre. "Argh", criez-vous, quand vous diagnostiquez enfin le problème par vous-même quelques heures plus tard. "Pourquoi le compilateur ne m'a-t-il pas simplement dit ce que c'était ?" Une façon d'éviter une

perte de temps pour le débogage de votre programme consiste à fournir des définitions pour des fonctions virtuelles pures qui ne devraient jamais être appelées, et à introduire de vrais codes malfaisants dans ces définitions qui vous feront immédiatement savoir si vous les avez appelées par accident. Exemple :

```
class B {
public:
    virtual bool f() = 0;
};

bool B::f() { // ceci ne devrait JAMAIS être appelé
    if( PromptUser( "pure virtual B::f called --
"
                    "abort or ignore?" ) == Abort
)
    DieDieDie();
}
```

Dans la classique fonction DieDieDie(), faites ce qu'il faut sur votre système pour accéder au débogueur, ou afficher une pile, ou obtenir d'une manière ou d'une autre des informations de diagnostic. Voici quelques méthodes

courantes qui vous feront passer dans le débogueur sur la plupart des systèmes. Prenez celle que vous préférez :

```
void DieDieDie() { // écrit dans un pointeur
null
    memset( 0, 1, 1 );
}

void DieDieDie() { // une autre méthode style C
    assert( false );
}

void DieDieDie() { // retourne au dernier
"catch(...)"
    class LocalClass {};
    throw LocalClass();
}

void DieDieDie() { // pour les standardophiles
    throw logic_error();
}
```

Vous saisissez le concept. Soyez créatifs.

Retrouvez l'article de Herb Sutter traduit par l'équipe C++ de [developpez.com](#) en ligne : [Lien 16](#)

## OpenGL dans Qt5

Le support d'OpenGL dans Qt5 a été modifié pour mieux l'intégrer avec les nouveaux modules de Qt : QtQuick2 et Qt3D. Cet article présente les modifications apportées dans Qt5.

### OpenGL dans Qt4

Dans Qt4, les fonctionnalités d'OpenGL sont implémentées dans un module spécifique, QtOpenGL. Ce module était utilisé par différentes classes pour bénéficier de l'accélération matérielle. Il existe plusieurs méthodes pour activer l'accélération matérielle :

- pour activer par défaut l'utilisation de OpenGL, utiliser la ligne de commande "-graphicssystem opengl" ou la fonction `QApplication::setGraphicsSystem("opengl")` ;
- pour `QGraphicsView` (QtGraphics) ou `QDeclarativeView` (QtQuick), utiliser la fonction `setViewport(new QGLWidget)` ;
- on peut également utiliser `QPainter` directement sur un `QGLWidget` ;
- avec le QML Viewer, utilisez la commande "-opengl".

### L'organisation des modules

Dans Qt4, on a donc un problème dans l'organisation des modules. Beaucoup de classes de QtGui peuvent dépendre des classes appartenant à QtOpenGL alors que ce module dépend normalement de QtGui.

C'est pour cela que de nouvelles classes font leur apparition dans Qt5 et que les différents modules ont été réorganisés :

- les classes QtOpenGL appartiennent au module QtGui et fournissent les fonctionnalités de base permettant l'accélération matérielle pour toutes les classes de rendu de Qt ;
- la classe `QWidget` n'est plus le parent de toutes les classes de rendu. Cette classe et ses dérivées (`QGraphicsView`) sont transférées dans le module "widgets" ;
- QtQuick 2 utilise maintenant scenegraph à la place de `QWidget` et la classe `QDeclarativeView` devient `QQuickView` et OpenGL est utilisé par défaut ;
- QtOpenGL continue d'exister pour fournir la classe `QGLWidget` et ses dérivés. Le code OpenGL Qt4 est donc compatible avec Qt5 ;
- le module Qt3d fait son apparition pour fournir un moteur 3D plus avancé. Ce module est suffisamment important et fera donc l'objet d'un article spécifique.

Remarque : il faut faire attention de ne pas confondre le

module QtOpenGL, contenant les classes commençant par QGL, et le module QtGui, contenant les classes commençant par QOpenGL (sans t).

### Les classes de QtGui dans Qt5

Le module QtGui de Qt5 réutilise des classes existantes du module QtOpenGL de Qt4. Les noms sont explicites :

1. `QOpenGLBuffer` ;
2. `QOpenGLContext` ;
3. `QOpenGLFramebufferObject` ;
4. `QOpenGLFramebufferObjectFormat` ;
5. `QOpenGLShader` ;
6. `QOpenGLShaderProgram`.

Plusieurs nouvelles classes font leur apparition :

1. `QOpenGLContextGroup` : groupe de contextes OpenGL pouvant partager des ressources. Cette classe est gérée automatiquement par les objets `QOpenGLContext` ;
2. `QOpenGLFunctions` : fournit un accès indépendant de la plateforme aux fonctions d'OpenGL ;
3. `QOpenGLPaintDevice` : permet de dessiner dans un contexte OpenGL avec `QPainter` ;
4. `QOpenGLStaticContext` : manque de documentation ;
5. `QOpenGLContextData` : manque de documentation ;
6. `QWindowsGLContext` : manque de documentation.

### Que faut-il modifier pour utiliser OpenGL dans Qt5

Tout d'abord, il faut Qt5. Le plus simple est d'utiliser les dépôts PPA sur Ubuntu : <https://launchpad.net/~forumnokia/+archive/fn-ppa>. Qt5 sera installé dans le répertoire /opt/qt5. Pour ajouter cette version de Qt dans QtCreator, il faut aller dans le menu "Outils" puis "Options...", allez dans "Compiler et exécuter..." puis l'onglet "Versions de Qt". Cliquer sur "Ajouter" et aller chercher le fichier "/opt/qt5/bin/qmake". Voilà, Qt5 est prêt à être utilisé.

Normalement, vous pouvez utiliser vos projets directement, mais il est préférable de prendre l'habitude d'ajouter le module "widgets" (remarque : l'ajout des modules core et gui n'est pas obligatoire puisqu'ils sont inclus par défaut) :

```
QT += core gui opengl
greaterThan(QT_MAJOR_VERSION, 4) {
    QT += widgets
}
```

Amusez-vous bien !

Retrouvez ce billet blog de Guillaume Belz en ligne : [Lien 17](#)

### L'utilisation des QGraphicsScene

Dans ce tutoriel, on va voir comment utiliser la classe `QGraphicsItem`, qui permet de façon générale d'intégrer des objets dans une scène 2D mais aussi de leur donner un aspect 3D, moyennant quelques astuces. Attention toutefois, ces outils de Qt ne sont pas non plus conçus pour afficher des scènes en 3D, ils ont simplement une bonne souplesse qui permet de réaliser de belles interfaces assez évoluées. On abordera donc l'intégration d'objets dans une scène en deux dimensions grâce à Qt, avant de montrer comment simuler la profondeur afin de donner une impression de 3D à ces objets.

#### 1. Introduction

Dans ce tutoriel, on va voir comment utiliser la classe `QGraphicsItem`, qui permet de façon générale d'intégrer des objets dans une scène 2D mais aussi de leur donner un aspect 3D, moyennant quelques astuces. Attention toutefois, ces outils de Qt ne sont pas non plus conçus pour afficher des scènes en 3D, ils ont simplement une bonne souplesse qui permet de réaliser de belles interfaces assez évoluées.

On abordera donc l'intégration d'objets dans une scène en deux dimensions grâce à Qt, avant de montrer comment simuler la profondeur afin de donner une impression de 3D à ces objets.

#### 2. Insérer des widgets dans une scène

##### 2.1. Quelques classes utiles

Afin de visualiser des widgets (et plus généralement des objets graphiques) dans une scène, il faut commencer par en créer une qui va contenir tout ce beau monde. La classe fournie par Qt s'appelle `QGraphicsScene` et permet d'insérer des `QGraphicsItem`. La classe abstraite `QGraphicsItem` représente tout ce qui est affichable dans une `QGraphicsScene`. On trouve notamment un nombre important de classes permettant de gérer des polygones, des textes, des `QPixmap`, des widgets...

La classe `QGraphicsProxyWidget`, héritant de `QGraphicsItem`, est particulièrement adaptée à l'insertion des widgets dans une `QGraphicsScene`. Il suffit de créer votre `QGraphicsProxyWidget`, de lui attacher le widget que vous désirez intégrer à la scène puis d'ajouter le proxy en tant qu'objet. Vous manipulez donc votre widget à travers le proxy, tout ceci étant transparent une fois que le proxy et le widget sont attachés. Il est de plus tout à fait possible d'intégrer des widgets complexes, comme une `QTabWidget`, qui contient elle-même d'autres widgets.

Dans les exemples, j'appliquerai les notions de ce tutoriel à des widgets, car c'est leur intégration qui m'a poussé à utiliser ces classes. Cependant, les méthodes utilisées sont pour la plupart héritées de `QGraphicsItem`, on peut donc les adapter à d'autres objets.

Enfin, comme la scène n'est pas un widget mais un simple conteneur, il faut aussi créer une vue avec la classe `QGraphicsView`, qui permet d'afficher le contenu de la scène. Sans plus attendre, on commence par créer la première scène, avec un bouton au milieu.

```
#include <QApplication>
#include <QtGui>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QPushButton *bouton = new QPushButton("Mon
bouton entre en scène !");
    QGraphicsScene scene;
    QGraphicsProxyWidget *proxy = new
QGraphicsProxyWidget();
    proxy->setWidget(bouton);
    scene.addItem(proxy);

    QGraphicsView view(&scene);
    view.show();

    return app.exec();
}
```



Résultat.

Ce code devrait afficher un bouton au milieu d'un écran blanc. Vous venez d'intégrer un premier widget dans une `QGraphicsScene`.

##### 2.2. Déplacer les objets dans la scène

Une fois le widget intégré à la scène à travers le proxy, Qt offre un grand nombre de méthodes pour modifier son positionnement.

###### 2.2.1. Les coordonnées

Si les systèmes de coordonnées 2D ne vous sont pas familiers, sachez que l'axe X est l'axe horizontal orienté vers la droite et que l'axe Y est l'axe vertical orienté vers le bas. *A priori*, si vous avez déjà un peu manipulé Qt ou la SDL par exemple, vous devriez vous y retrouver.

Il faut de plus savoir que la vue, la scène et les items ont chacun leur propre système de coordonnées. Ainsi, le point (0, 0) qui est le point en haut à gauche de la vue dans son propre système aura des coordonnées différentes dans le système de coordonnées de la scène. Ceci se modélise par

le schéma suivant, issu de la documentation, et dans lequel le repère de la vue est en vert, la scène est en bleu et l'item en rouge.

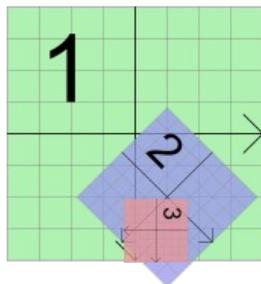


Image issue de la documentation officielle Qt ([Lien 18](#)).

Afin de simplifier les conversions d'un système de coordonnées à un autre, Qt fournit des méthodes `mapToScene()` et `mapFromScene()` dans les classes `QGraphicsView` et `QGraphicsItem` qui permettent de retrouver les coordonnées d'un point dans la scène à partir des coordonnées dans la vue ou l'item et inversement.

### 2.2.2. Translation

La fonction `setPos()` de `QGraphicsItem` prend deux paramètres, `dx` et `dy`, qui sont les déplacements relatifs respectivement sur l'axe X et l'axe Y de la scène, paramètres qui deviendront alors les nouvelles coordonnées de notre objet. À savoir qu'une méthode surchargée existe, prenant en paramètre un `QPointF`. À vous de voir laquelle vous préférez. Il est aussi possible d'appeler la fonction `move()`, qui ajoute le déplacement sur les deux axes à la position actuelle de l'objet. On essaie tout de suite de déplacer notre objet dans la scène :

```
#include <QApplication>
#include <QtGui>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QPushButton *bouton = new QPushButton("Mon
bouton entre en scène !");
    QGraphicsScene scene;
    QGraphicsProxyWidget *proxy = new
QGraphicsProxyWidget();
    proxy->setWidget(bouton);
    scene.addItem(proxy);

    proxy->setPos(150, 200);

    QGraphicsView view(&scene);
    view.show();

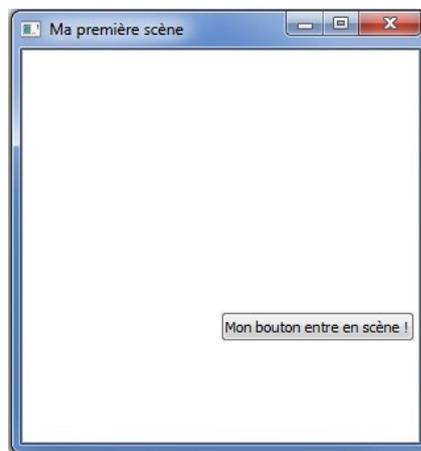
    return app.exec();
}
```

En exécutant ce code, on se rend compte que le widget ne s'est pas déplacé. En effet, c'est une des particularités de la `QGraphicsView`, qui s'arrange par défaut pour placer les objets au milieu de la vue.

Heureusement, il est possible de définir la zone de la scène que l'on désire voir à l'écran. Ajoutez cette ligne après la création de la scène.

```
scene.setSceneRect(-150, -150, 300, 300)
```

Les deux premiers nombres sont les coordonnées du point qui doit être placé dans le coin supérieur gauche de la vue (ici, `(-150, -150)`). Les deux nombres suivants sont la largeur et la hauteur de la scène que je désire afficher. Ainsi, comme le point `(-150, -150)` est le coin supérieur gauche de ma fenêtre, on retrouve l'origine de la scène au centre de l'écran. En relançant votre code avec et sans le déplacement du bouton, on voit cette fois-ci une différence de position.



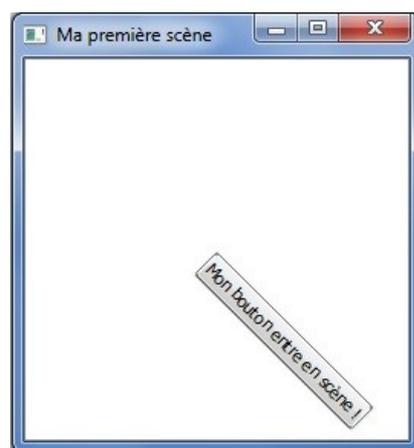
Résultat.

### 2.2.3. Rotation

Afin d'appliquer une rotation à l'objet, il suffit d'utiliser la méthode `setRotation()`, pour laquelle vous devrez fournir l'angle de rotation en degrés. Un angle positif tourne l'objet dans le sens des aiguilles d'une montre, un angle négatif le tourne dans le sens inverse.

Par exemple :

```
proxy->setRotation(45);
```

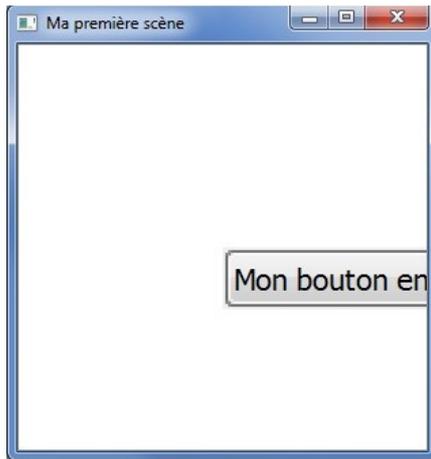


Résultat

### 2.2.4. Mise à l'échelle

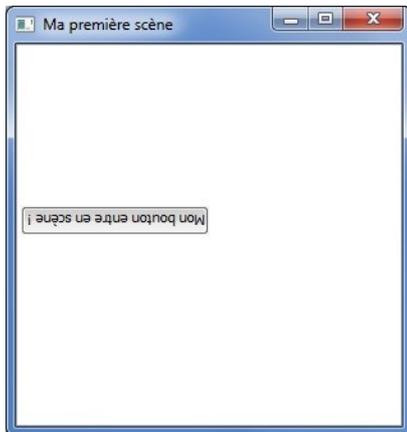
Enfin, la méthode `setScale()` permet d'augmenter ou de diminuer la taille prise par l'objet dans la scène. Un nombre supérieur à 1 agrandit l'image, tandis qu'un nombre compris entre 0 et 1 la réduit. Ci-dessous, on a pris un facteur de 2 pour agrandir le bouton, dont une partie sort maintenant de l'écran.

```
proxy->setScale(2);
```



Résultat

Cette fonction prend aussi en compte les nombres négatifs, ce qui permet d'effectuer une symétrie de l'image en plus de modifier sa taille. Avec un facteur -1 on obtient ainsi un bouton renversé.



Résultat.

### 2.2.5. Conclusion

Ces opérations de base sont en gros le minimum pour pouvoir gérer vos objets dans la scène 2D.

On parle ici de scène 2D puisque l'on n'a parlé que de deux axes pour la translation et uniquement d'une rotation autour de l'axe perpendiculaire à l'écran. En effet, un QGraphicsItem est bien affiché en deux dimensions dans la scène et ne prend pas directement la profondeur en paramètre.

Avant de donner pour de bon de la profondeur à notre scène et de faire des manipulations 3D, il va falloir passer un peu de temps sur les matrices. Le calcul matriciel est un outil mathématique qui permet d'effectuer les manipulations de base en 3D et pour lequel Qt donne tous les outils nécessaires pour limiter les calculs à faire.

## 3. Les transformations avec QTransform

On a vu que les QGraphicsItem proposaient diverses méthodes pour déplacer, tourner ou agrandir les widgets insérés. Seulement, tout cela n'a pas vraiment l'air d'avoir la profondeur qu'on attend d'une scène 3D. Pour cela, on

va utiliser une nouvelle classe de Qt, la classe QTransform.

Si vous avez travaillé un peu les mathématiques dans vos études supérieures, ce qui suit devrait vous être familier, dans le cas contraire, lisez attentivement la suite.

### 3.1. Utiliser un QTransform avec un QGraphicsItem

Un QTransform représente un objet mathématique appelé matrice, que l'on peut représenter dans notre cas comme un tableau comportant trois lignes et trois colonnes et contenant toutes les informations nécessaires pour effectuer des modifications sur les coordonnées d'un objet dans l'espace.

Voici comment on représente généralement une matrice :

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

L'édition d'un QTransform est très simple dans le code, puisque Qt s'occupe de toute la partie mathématique, en proposant des fonctions permettant de générer et modifier ces matrices.

Comme un exemple vaut mieux qu'un long discours, voici tout de suite la manière de réaliser une translation avec un QTransform.

```
#include <QApplication>
#include <QtGui>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QPushButton *bouton = new QPushButton("Mon
bouton entre en scène !");

    QGraphicsScene scene;
    scene.setSceneRect(-150, -150, 300, 300);

    QGraphicsProxyWidget *proxy = new
QGraphicsProxyWidget();
    proxy->setWidget(bouton);
    scene.addItem(proxy);

    QTransform matrix;
    matrix.translate(150, 200)
    proxy->setTransform(matrix);

    QGraphicsView view(&scene);
    view.show();

    return app.exec();
}
```

Comme précédemment, il a suffi de renseigner les valeurs de mon vecteur déplacement et Qt a généré tout seul la matrice correspondante.

Exécutez ce code qui doit vous afficher la même chose que tout à l'heure. Oui, en l'occurrence, le résultat est le même lorsque l'on regarde la fenêtre. Le widget s'est bien déplacé

de 150 pixels vers la droite et de 200 pixels vers le bas.

De même, on peut générer des rotations et des mises à l'échelle à l'aide des méthodes `rotate()` et `scale()` de la classe `QTransform`.

L'avantage des `QTransform` par rapport à l'utilisation des méthodes précédentes est la possibilité de faire des transformations composées en une seule opération, plutôt que d'appliquer une méthode à la fois. En revanche, il n'est pas souhaitable de combiner l'utilisation des `QTransform` avec la méthode `move()` du `QGraphicsItem` par exemple, puisque les modifications effectuées par un `QTransform` ne sont récupérables que par la méthode `transform()` du `QGraphicsItem` modifié. Cela veut dire que les coordonnées de l'objet, auxquelles on pouvait accéder par `QGraphicsItem::x()` et `QGraphicsItem::y()`, n'ont pas été modifiées et valent toujours 0 !

```
QTransform matrix;
matrix.translate(150, 200);
proxy->setTransform(matrix);

matrix = QTransform().translate(proxy->x(),
proxy->y())
proxy->setTransform(matrix);
```

En remplaçant le code précédent par ces lignes, on se rend compte que le widget n'a pas bougé de l'origine de la scène. En effet, la dernière transformation étant celle qui prend en compte les paramètres `x()` et `y()` du proxy, il est revenu à la position (0, 0) car l'application du `QTransform` ne les a pas modifiés. Il faut donc éviter de mélanger les outils et préférer celui-ci qui semble convenir au besoin.

Le `QTransform` a donc plutôt pour but de modifier la façon dont est affiché l'objet, que de modifier l'objet lui-même.

## 3.2. Composer les matrices

### 3.2.1. Un outil pratique

Maintenant qu'on a vu comment utiliser les matrices avec les objets de la scène, voyons pourquoi cette nouvelle notion a été introduite : c'est un outil très puissant, souvent utilisé lorsque l'on travaille en trois dimensions.

Une matrice (et donc ici un `QTransform`) ne se limite pas à effectuer une seule opération à la fois. Elle peut en effet contenir les informations nécessaires pour effectuer à la fois une rotation, une translation et une mise à l'échelle ! Il ne suffira plus que d'appliquer la matrice à l'objet avec la méthode `QGraphicsItem::setTransform()` et le tour est joué.

Mathématiquement, la composition de matrices s'obtient en les multipliant entre elles, pour donner une nouvelle matrice qui contient les deux transformations. Qt permet de faire ces transformations directement via des méthodes de la classe `QTransform`.

Par exemple, si on veut créer une matrice de rotation puis ajouter une translation, on écrit ceci :

```
QTransform matrix;
matrix.rotate(30, Qt::YAxis);
matrix.translate(20, 15);
```

Il est aussi possible d'utiliser directement la multiplication pour composer vos matrices. Cependant il faut faire attention à l'ordre des calculs. La matrice la plus à gauche dans la multiplication correspond à la transformation effectuée en dernier. Les `QTransform` tiennent compte de cela, l'utilisation des méthodes `rotate()` et `translate()` place la matrice paramètre au début du calcul et non à la fin. Par exemple, voici deux lignes de code qui effectuent le même calcul.

```
matrix.rotate(30, Qt::YAxis);
matrix = QTransform::rotate(30, Qt::Axis) *
matrix;
```

Par conséquent, soyez encore plus prudent dans le choix de l'ordre des lignes de code.

Si vous utilisez les multiplications directement, placez vos matrices dans l'ordre inverse de celui dans lequel vous devez faire vos transformations. Si vous utilisez les méthodes fournies par Qt, vous devrez les écrire dans l'ordre normal, ce qui est plus intuitif.

Il suffit alors d'appliquer la matrice à l'objet situé dans la scène pour que ces deux transformations soient prises en compte.

### 3.2.2. Une question d'ordre

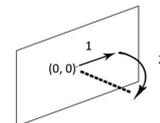
Voyons maintenant pour quelle raison l'ordre des transformations est important et doit être bien choisi. Par exemple, les deux codes suivants donneront deux matrices (et donc deux transformations) différentes.

```
QTransform matrix;
matrix.translate(10, 10);
matrix.rotate(45, Qt::YAxis);
```

```
QTransform matrix;
matrix.rotate(45, Qt::YAxis);
matrix.translate(10, 10);
```

Avec le premier code, on a déplacé l'objet avant de le faire tourner autour de l'origine ; dans le second, on l'a déplacé après l'avoir fait tourner autour de l'origine.

Dans le premier cas, la transformation finale pourrait se schématiser ainsi :



L'objet a parcouru le chemin en pointillés et ne se trouve ainsi plus dans le plan (x, y) ! L'autre cas, en revanche, correspond bien à ce qu'on attend intuitivement, c'est-à-dire un objet situé dans le plan (x, y) mais qui a tourné sur lui-même.

Ainsi, pour orienter un objet dans la scène puis le placer, il faut effectuer d'abord la rotation suivant l'angle voulu, puis le déplacement. La seconde solution est donc celle vous devriez utiliser.

## 4. Des widgets en 3D

### 4.1. Rotation autour de l'axe Y

On va tout d'abord voir comment appliquer aux widgets des rotations autour de l'axe Y. Un petit tour dans la documentation apprend que la méthode `QTransform::rotate()`, qui prend en paramètre un angle et un axe, correspond tout à fait à ce besoin.

Voyons ce que ça va donner avec un `QWebView`, par exemple (n'oubliez pas d'ajouter la ligne `QT += webkit` à votre fichier `.pro`).

```
#include <QApplication>
#include <QtGui>
#include <QWebView>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QWebView *web = new QWebView();
    web->load(QUrl("http://www.developpez.com"));
    web->show();

    QGraphicsScene scene;
    scene.setSceneRect(0, 0, 1000, 800);

    QGraphicsProxyWidget *proxy = new
    QGraphicsProxyWidget();
    proxy->setWidget(web);
    scene.addItem(proxy);

    QTransform matrix;
    matrix.rotate(45, Qt::YAxis);
    proxy->setTransform(matrix);

    QGraphicsView view(&scene);
    view.show();

    view.setWindowTitle("Ma première scène");
    return app.exec();
}
```

Compilez ça et normalement vous obtenez une petite perspective sur votre widget, dont les fonctionnalités sont toujours utilisables ! Un `QTextEdit` continuera à être éditable dans l'espace, une `QWebView` continuera à charger les pages demandées, etc. Bien sûr, on peut se contenter de formes géométriques plus basiques, mais il est bon de savoir que cela fonctionne avec tous les widgets, il est donc possible d'afficher une page Web, par exemple.



### 4.2. Implémenter le déplacement

Maintenant, on peut ajouter un peu d'interaction à la scène, en permettant à notre vision de bouger par rapport aux objets. Pour cela, on crée une classe spéciale, héritée de `QGraphicsProxyWidget`, qui va contenir les informations dont on souhaite garder la trace : orientation, position, échelle...

Voici l'en-tête de la classe proposée.

```
#include <QGraphicsProxyWidget>

class MyProxy : public QGraphicsProxyWidget
{
public:
    MyProxy();

    qreal rotationY();
    void setRotationY(qreal rotation);

    QPointF center();

private:
    qreal m_rotationY; // enregistre la rotation
    // autour de l'axe Y
    QPointF m_center; // contient la position du
    // centre du widget
};
```

Puis on crée une classe dérivée de `QGraphicsScene` pour gérer les événements de la souris. Je vous laisse créer la classe tout seul, il suffit de réimplémenter la méthode `mouseMoveEvent()`, appelée en cas de déplacement de la souris et de créer un bouton dans le constructeur que l'on attache à une instance de `MyProxy` placée en attribut de `MyScene`.

```
MyScene::MyScene() : QGraphicsScene()
{
    QPushButton *bouton = new QPushButton("Mon
    bouton entre en scène !");
    m_proxy = new MyProxy();
    m_proxy->setWidget(bouton);
    this->addItem(m_proxy);
}
```

On va, par exemple, demander d'effectuer une rotation autour de l'axe Y lorsque la souris est déplacée latéralement et que le bouton gauche est enfoncé.

```
void
MyScene::mouseMoveEvent(QGraphicsSceneMouseEvent*
e)
{
    if(e->buttons() & Qt::LeftButton)
    {
        QPointF delta(e->scenePos() - e-
        >lastScenePos());
        qreal rotation = delta.x();
        m_proxy->setRotationY(rotation + m_proxy-
        >rotationY());

        QTransform matrix;
        matrix.rotate(m_proxy->rotationY(),
        Qt::YAxis);
        m_proxy.setTransform(matrix);
    }
}
```

Dans la fonction main il suffit maintenant de créer une instance de MyScene et de lui associer la vue.

```
int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    MyScene scene;
    scene.setSceneRect(-150, -150, 300, 300);

    QGraphicsView view(&scene);
    view.setWindowTitle("Ma première scène");
    view.show();
    return app.exec();
}
```

Maintenant, en déplaçant la souris avec le bouton gauche enfoncé, le widget devrait se mettre à tourner verticalement autour de l'origine de la scène. Cependant, cette origine coïncide avec le bord gauche du widget, celui-ci ne tourne donc pas sur lui-même.

Par conséquent, pour remédier à ce problème, il faut déplacer le widget pour que son centre se situe au point (0, 0) ! Une autre solution consisterait à utiliser la méthode QGraphicsItem::setTransformOriginPoint() et à envoyer le centre de l'item en paramètre.

On reprend la scène avec la QWebView. Le déplacement à effectuer est représenté par le schéma ci-dessous : pour que la rotation s'effectue par rapport au centre du widget, il faut ramener ce centre à l'origine de la scène. C'est donc le déplacement représenté par la flèche rouge qui va réaliser cela.



*A priori*, vous devriez savoir faire cela tout seul. Il suffit de composer notre matrice d'une translation supplémentaire pour déplacer notre widget vers la gauche et en haut, sur une longueur respective de la moitié de la largeur et la moitié de la hauteur (avec un signe négatif, car on va vers la gauche et vers le haut). Vous pouvez récupérer la taille de vos widgets avec les méthodes proxy->widget()->width() et proxy->widget()->height() et les ajouter comme attributs de votre classe MyProxy pour ne plus avoir à les rechercher. Voici donc le code de la rotation avec le recentrage du widget.

```
QTransform matrix;
matrix.rotate(m_proxy->rotationY(), Qt::YAxis);
matrix.translate(-m_proxy->widget()->width()/2,
-m_proxy->widget()->height()/2);
m_proxy->setTransform(matrix);
```

Pensez à déplacer l'origine de la scène au centre de la fenêtre avec la méthode QGraphicsScene::setSceneRect() si ce n'est déjà fait, ou bien votre widget sera coupé par le bord de l'écran à cause de ce décalage.

```
scene.setSceneRect(-150, -150, 300, 300);
```

## 5. Pour aller plus loin : gestion de la caméra

### 5.1. La caméra : reprenons à l'envers !

On a déjà dit que l'ordre de composition des transformations était important. Ça le devient encore plus pour une caméra dans l'espace !

Prenons un exemple. Considérons la tourelle d'un tank dans la scène. Pour la placer correctement, vous allez devoir lui appliquer sa rotation par rapport au tank, puis la rotation du tank par rapport à la scène, puis enfin la translation qui la positionnera à l'endroit voulu.

Si maintenant vous voulez placer une caméra (votre point de vue) dans la scène, que se passe-t-il ? Dans ce cas, si vous tournez sur vous-même, vous voyez le monde 3D entier qui tourne autour de votre position. La rotation de la caméra doit donc être appliquée après sa translation.

Autant vous prévenir tout de suite, cet exemple ne vous permettra pas de tourner sur vous-même et de vous déplacer dans l'espace, mais plutôt de déplacer tous les objets de la scène d'un coup ou bien de tous les faire tourner devant vous. La construction des matrices utilisées dans les jeux vidéo, par exemple pour afficher correctement la scène devant la caméra, n'est pas évidente à faire à la main et dépasse le cadre de ce tutorial.

Voyons donc tout de suite concrètement comment on peut donner l'impression que l'on se déplace dans la scène.

### 5.2. Créer une caméra

Déterminons tout de suite de quelles informations nous allons principalement avoir besoin pour notre caméra : position dans la scène et orientation, tout comme les objets de notre scène.

```
#include <QPointF>

class Camera
{
public:
    Camera();

    QPointF pos();
    qreal zpos();
    qreal rotationY();

    void setPos(QPointF pos);
    void setZPos(qreal pos);
    void setRotationY(qreal angle);

private:
    QPointF m_pos;
    qreal m_zpos;
    qreal m_rotationY;
    MyScene *m_scene;
};
```

Vous remarquerez aussi que j'ai ajouté un attribut `m_zpos` à la caméra.

C'est en fait une astuce que je voulais vous montrer et qui permettra de donner l'impression que l'on déplace celle-ci vers l'avant où l'arrière. Nous verrons comment faire lors de l'implémentation dans la scène. Sachez simplement que la valeur de cet attribut est modifiée lors de l'appui d'une touche, par exemple des flèches haut et bas ou bien la molette de la souris, au choix.

Ainsi, il n'y a pas grand-chose de plus à implémenter pour avoir une caméra de base. Notez qu'ici je ne considère qu'une rotation par rapport à l'axe Y, mais libre à vous d'ajouter les deux autres axes si vous le voulez.

L'avantage d'utiliser un objet caméra dans cette scène, c'est que vous n'avez que les paramètres de ladite caméra à modifier lorsque vous la tournez ou la déplacez, plutôt que de redéplacer tous les objets un par un.

```
QList<MyProxy *> m_objets;
```

Pour ajouter un objet à la liste, c'est très simple, vous pouvez utiliser l'opérateur de flux `<<`.

```
MyProxy *proxy;  
// ...  
m_objets << proxy;
```

Lorsque l'on modifie un paramètre de la caméra comme la rotation, on appelle une méthode `updateScene()` de la classe `MyScene` qui va parcourir tous les objets de la scène et appeler, pour chacun d'eux une méthode `replacer()` se chargeant d'appliquer les transformations dues à la caméra.

```
void MyScene::updateScene()  
{  
    foreach (MyProxy *objet, m_objets)  
    {  
        objet->replacer();  
    }  
}
```

Il suffit maintenant d'ajouter un attribut caméra dans la classe `MyProxy` et d'écrire la fonction `replacer()` comme suit.

```
void MyProxy::replacer()  
{  
    QTransform m;  
    m.translate(this->center().x(), this->center().y());  
    m.translate(-this->widget()->width()/2,  
-this->widget()->height()/2);  
  
    // ajout des transformations liées à la position de la caméra  
    m.translate(-m_camera.pos().x(),  
-m_camera.pos().y());  
    m *= QTransform().rotate(m_camera.yaw(),  
Qt::YAxis);  
  
    // simulation du déplacement suivant Z  
    qreal scale = m_camera.zpos();  
    m *= QTransform().scale(scale, scale);  
  
    this->setTransform(m);  
}
```

Enfin vous remarquez l'utilisation de la fonction `QTransform::scale()` qui a un paramètre dépendant de l'attribut `m_zpos` de ma caméra. En effet, si je considère que lorsque ma position en Z augmente je me rapproche de la scène, alors cela revient à voir les objets dans la scène en plus gros, donc à les agrandir ! Ici j'ai fait une implémentation assez basique pour que vous compreniez le principe, on peut cependant affiner le comportement en ajoutant des coefficients multiplicateurs pour gagner en précision.

## 6. Conclusions

Voilà ! J'espère que les classes `QGraphics****` de Qt vous paraissent plus familières et que vous avez une idée du genre de scènes qu'il est possible de réaliser.

Pour des notions plus exhaustives, vous pouvez de plus visiter la documentation sur ce module de Qt : [Lien 19](#).

Retrouvez l'article de Julien Chichignoud en ligne : [Lien 20](#)



### Automatisez sous Mac Automator travaille pour vous



Automatiser... oui, mais pourquoi ? Pour ne plus effectuer de tâches ennuyeuses et répétitives, mais également pour simplifier l'utilisation du Mac, standardiser des tâches récurrentes ou des procédures, fournir aux utilisateurs d'un même réseau des outils homogènes.

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.

Dans cet ouvrage, Sylvain Gamel vous présente Automator, son interface, son fonctionnement. Applications, services et modules d'impression sont trois moyens très pratiques pour mettre en place des automatismes dans vos procédures de travail. Vous apprendrez ce qu'est un processus, élément de base d'Automator, et comment en créer un. Vous saurez, par exemple, comment construire une planche contact. L'auteur explique également ce qu'est un service et vous pourrez alors manipuler vos fichiers et dossiers. Vient ensuite la description des actions de dossier qui permettent aux processus de s'exécuter seuls. Un long chapitre est consacré aux actions disponibles depuis les applications de votre Mac. Vous disposerez ainsi des outils pour construire vos propres processus et services. Le chapitre suivant vous permettra de mettre en œuvre ces actions pour créer des processus qui s'intégreront dans le système d'impression du Mac. Pour finir, vous verrez comment contourner les

limitations d'Automator et publier vos propres actions. Un aperçu de ce qu'AppleScript et les scripts Unix pourraient vous apporter conclut le livre. Apprendre à utiliser Automator, c'est un peu comme apprendre à faire du vélo. Le meilleur moyen reste la pratique. L'auteur vous accompagne donc tout au long de l'ouvrage et vous montre comment maîtriser Automator à l'aide de très nombreux exemples et explications détaillées. Si vous souhaitez gagner du temps et mieux utiliser votre Mac, n'attendez plus et découvrez l'univers d'Automator.

#### Critique du livre par Aurélien Gaymay

Ce livre est fait pour tous les niveaux : que vous soyez débutant ou expert, vous apprendrez quelque chose.

Grâce à ce livre, vous allez pouvoir contrôler et dompter votre Mac sans aucune connaissance en programmation. Pas besoin d'avoir des années d'expérience sur Mac OS pour réaliser un programme qui vous facilite la vie en évitant les tâches répétitives.

Avec Automator, faite de votre Mac un assistant qui vous permettra de générer des mails automatiquement, de traiter des images, chiffrer des documents PDF ou de créer des outils de manipulations complexes de dossier.

Pour les développeurs connaissant le langage AppleScript, cela leur permettra d'éviter d'écrire des lignes de code pour certaines tâches, mais aussi d'optimiser au maximum les ressources d'Automator en l'associant à AppleScript, mais aussi à des langages comme Python ou Perl.

Faites de votre Mac un super assistant en le faisant travailler à votre place...

*Retrouvez cette critique de livre sur la page livres Mac : [Lien 21](#)*

### Application et méthode d'envoi de lettres d'information en CDO

Application Access destinée à l'envoi de lettres d'information (newsletters) en CDO.

#### 1. Introduction



Je vous propose une petite application d'envoi de lettres d'information (newsletters).

Cette application fonctionnelle est essentiellement destinée à démontrer qu'Access permet de concevoir des outils sympathiques en fonction des besoins.

Elle pourra être utilisée en tant que telle par une association par exemple, mais servira surtout de support aux développeurs débutants.

En effet, outre l'aspect messagerie, l'application utilise de nombreuses solutions proposées dans la Faq ([Lien 22](#)) ou le Forum ([Lien 23](#)).

Afin de pouvoir bénéficier des options de texte enrichi, l'application a été développée en Accdb et ne peut donc être utilisée qu'avec Access 2007 ou plus. Si cette base est convertie en Mdb, cette option ne pourra pas être utilisée.

Vous pouvez également vous en servir avec le Runtime, mais dans ce cas le code ne sera pas accessible.

#### 2. Applicatif

Vous pouvez télécharger le fichier ici : [Lien 24](#).

#### 3. Quelques principes et règles sur les envois de newsletters

Ce type de logiciel est destiné à l'envoi en nombre. C'est ce que l'on appelle le mass mailing.

Il est soumis à la réglementation, car s'apparentant à du e-mail publicitaire.

En France, la loi stipule que les internautes doivent donner leur accord ou s'inscrire pour recevoir des e-mails publicitaires.

Un envoi publicitaire non désiré peut s'apparenter à un spam ([Lien 25](#)).

Cela concerne les envois de campagnes publicitaires, mais également les lettres d'information, etc.

Tout e-mail devra contenir un lien de désabonnement, ou la procédure à suivre pour ne plus recevoir de messages.

Vous pouvez avoir les renseignements précis sur le site de la CNIL.

À noter également que les fournisseurs d'accès peuvent limiter les envois à partir de leurs sites, ou au contraire refuser de recevoir ceux provenant d'adresses e-mail ou Ip considérées comme black-listées ([Lien 26](#)).

En conclusion ce type d'outil, peut servir à une petite association avec des envois ponctuels, et non à une grosse structure ayant des milliers d'envois par semaine.

Dans ce cas il vaut mieux passer par une entreprise spécialisée.

#### 4. Possibilités du logiciel

##### Fonctionnalités du logiciel :

- envoyer des newsletters à partir de plusieurs serveurs ;
- avoir autant de listes d'abonnés (listes de diffusion) que souhaité ;
- importer des listes de diffusion hébergées dans une base MySQL d'un serveur Web (Free dans l'exemple) ;
- saisir les newsletters en texte enrichi ;
- copier le contenu des newsletters à partir de sources HTML provenant d'un logiciel tiers ;
- visualiser ce code HTML ;
- pouvoir insérer une pièce jointe ;
- ajouter un éventuel lien vers une page Web ;
- ajouter un éventuel lien de désinscription ;
- prévoir un nombre maximum d'envois ;
- conserver les messages ;
- suivre l'envoi à l'aide d'une barre de progression ;
- faire une reprise d'envoi après incident ou si le quota d'envois est atteint ;
- garder un historique des envois.

#### 5. Notice d'utilisation

##### 5.1. Paramètres généraux

##### 5.1.1. Paramétrage des connexions

Ces paramètres sont dans la majorité des cas ceux donnés par votre fournisseur d'accès (FAI : [Lien 27](#)) ou votre prestataire de messagerie.

Si vous avez un logiciel de messagerie sur votre PC, vous pouvez récupérer les paramètres SMTP de votre compte de messagerie.

Bien lire le contrat de votre prestataire pour savoir si votre nombre d'envois quotidiens est limité.

S'il l'est, il faut remplacer le paramètre "**Nb mails maxi/jour**" illimité (9999) par celui qui vous est indiqué, sur lequel vous déduirez un nombre moyen d'autres messages.

Par exemple si votre nombre est limité à 100, notez 90 pour avoir la possibilité d'envoyer une dizaine de messages classiques.

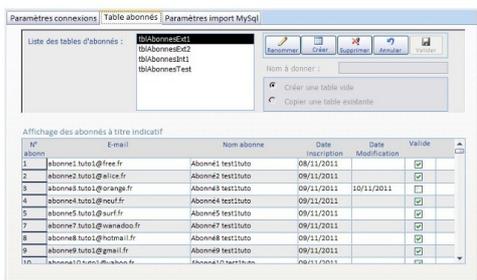
Pour modifier un paramètre, cliquez sur le numéro de connexion dans la liste. Le détail de la fiche apparaît plus haut. Faites les modifications et validez.

Vous pouvez bien sûr ajouter ou supprimer un paramétrage, avec les boutons adéquats.



## 5.1.2. Paramétrage des tables d'abonnés

Cet onglet va nous permettre de paramétrer les tables d'abonnés :



### 5.1.2.1. Principe

Nous pouvons créer autant de listes d'abonnés (ou listes de diffusion) que nous voulons.

Chaque création fait l'objet d'une table spécifique.

Ces tables sont reconnues comme tables d'adhérents car possédant un champ spécifique "**DateInscription**".

Il est conseillé de leur donner des noms facilement identifiables.

### 5.1.2.2. Renommer les tables

Vous avez la possibilité de renommer ces tables, si la logique ne vous convient plus.

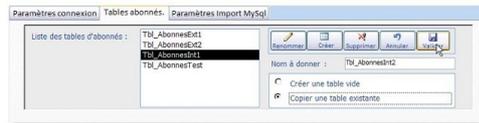
Pour cela :

- sélectionnez celle que vous voulez renommer dans la liste ;
- saisissez le nouveau nom dans la zone "Nom à donner" ;
- validez et passez éventuellement à la table suivante.

### 5.1.2.3. Copier une table

Vous pouvez créer de nouvelles listes de diffusion.

Vous pourrez le faire soit en créant une table vide (dans ce cas, seule la structure de la table sera créée), soit en recopiant une table existante et en lui donnant un autre nom.



Il est recommandé de créer une table de test, contenant une ou plusieurs adresses auxquelles vous avez accès, afin de simuler les envois et ainsi contrôler le contenu de vos messages.

### 5.1.2.4. Paramétrages d'import MySQL

Il peut être intéressant, pour faciliter les choses, que les inscriptions soient faites directement par les abonnés à l'aide d'un formulaire en ligne, dont les données seront stockées dans une base MySQL hébergée sur le serveur du fournisseur d'accès.

Dans ce cas si la structure est compatible, on peut envisager avant chaque envoi d'importer les données enregistrées sur cette base. Pour ce faire, nous aurons besoin de différents paramètres correspondant au script PHP présent sur ce serveur.

Pour pouvoir utiliser cette option, il est nécessaire d'avoir un minimum de connaissances sur l'hébergement et les scripts.

#### 5.1.2.4.1. Script PHP

Dans un premier temps il vous faudra stocker sur le serveur un fichier PHP qui servira de passerelle avec Access.

#### Script PHP

```
<?PHP
$query=$_POST['sql'];

$link = mysql_connect("sql.free.fr",
"Username", "password");
mysql_select_db("sqldvp", $link) or
die(mysql_error());

mysql_query("SET NAMES 'utf8'");
$result = mysql_query($query, $link) or
die(mysql_error());

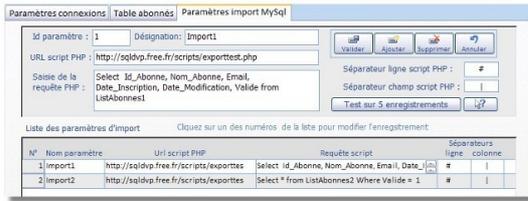
$fields = mysql_num_fields($result);
$rows=@mysql_numrows($result);

for ($ligne=0 ; $ligne<@mysql_numrows($result);
$ligne++)
{
for ($colonne = 0;$colonne < $fields ;
$colonne++)
{
$resultquery.= mysql_result($result, $ligne,
$colonne).'|'; //separateur colonne |
}
$resultquery.='#'; // séparateur line #
}
echo $resultquery;
mysql_close();
?>
```

Il faut remplacer vos propres paramètres de connexion dans la ligne "\$link = ...".

Il faudra également personnaliser vos propres séparateurs

de ligne et colonne ('|' et '#' dans notre exemple).  
 Bien noter l'emplacement de ce script sur le serveur.



Comme vous le constatez dans la vue ci-dessus, il faut saisir l'URL du script ainsi que les séparateurs mentionnés plus haut.

Une requête "Select" est nécessaire pour pouvoir sélectionner les données à importer.

Par exemple :

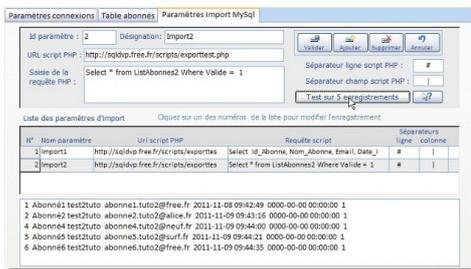
```
Select Id_Abonne, Nom_Abonne, Email,
Date_Inscription, Date_Modification, Valide from
ListAbonnes1
```

Ou encore :

```
Select * from ListAbonnes2 Where Valide = 1
```

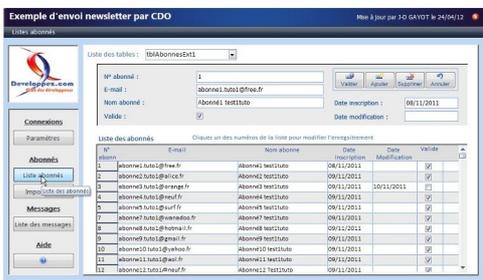
Pour vérifier que vos paramètres et que votre requête sont bien saisis cliquez sur le bouton "Test sur 5 enregistrements".

Si tout se passe bien et si votre table externe n'est pas vide, cinq enregistrements s'afficheront.



### 5.1.3. Tables d'abonnés

#### 5.1.3.1. Saisie classique

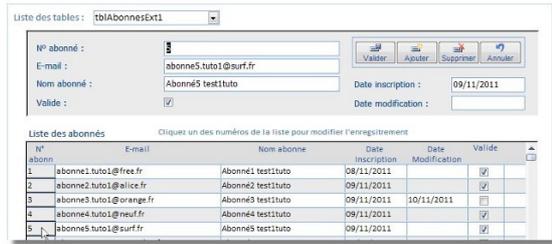


Vous pourrez dans ce formulaire saisir directement des données, en supprimer ou en corriger.

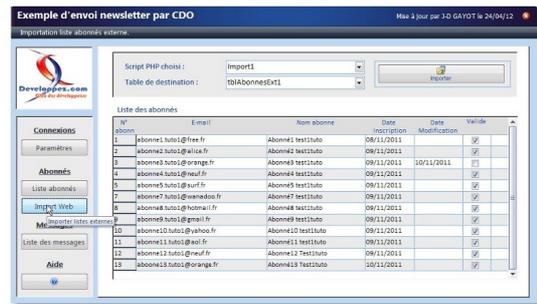
Vous pouvez également sélectionner une table dans la liste déroulante et afficher ainsi tous les abonnés correspondant à cette table.

Si vous voulez modifier les données d'un adhérent, cliquez sur son code dans la liste et la fiche correspondante s'affichera.

N'oubliez pas de valider après chaque saisie ou modification, pour que les changements soient effectifs.



#### 5.1.3.2. Saisie par import externe.



Vous pourrez au départ de ce menu, alimenter une table à partir des données saisies sur un serveur externe.

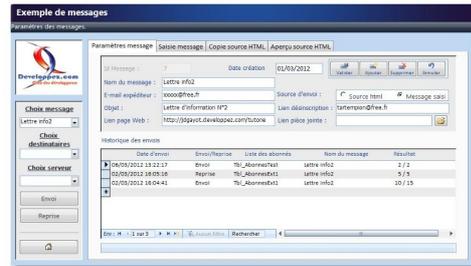
Il suffit de sélectionner un script prédéfini dans les paramètres, puis de choisir la table de destination.

Une demande de confirmation vous sera adressée avant le lancement de l'import.

Le fait d'importer des données supprime toutes celles que vous aviez auparavant dans la table choisie.

### 5.2. Messages

Vous accédez à ce formulaire par le bouton "Liste des messages" du menu général.



#### 5.2.1. Paramètres et contenu des messages

Vous pouvez à partir de la liste déroulante, sélectionner un des messages déjà créés et ainsi vérifier les paramètres de diffusion et l'historique des éventuels envois.

Vous pouvez également en saisir un nouveau, en supprimer, ou en modifier.

##### 5.2.1.1. Quelques explications

###### Nom du message :

donnez un nom représentatif.

###### Mail expéditeur :

donnez votre adresse mail d'expéditeur correspondant au serveur d'envoi que vous choisirez.

**Objet :**

donnez là aussi un nom représentatif. C'est celui qui apparaîtra dans le mail.

**Source d'envoi :**

choix correspondant à l'onglet dans lequel vous aurez saisi votre message.

**Lien page Web :**

correspond à l'URL d'un page Web sur laquelle le destinataire pourra visualiser le message si celui-ci ne s'affiche pas correctement dans sa messagerie.

Cette zone est facultative et est surtout destinée aux envois par l'option "source HTML".

**Lien désinscription :**

ce lien est normalement obligatoire dans le cas d'envois en nombre.

Il peut être soit une adresse e-mail si vous choisissez d'être informé par e-mail, soit par un lien Web vers par exemple un formulaire de désinscription lié à votre table externe.

**Lien pièce jointe :**

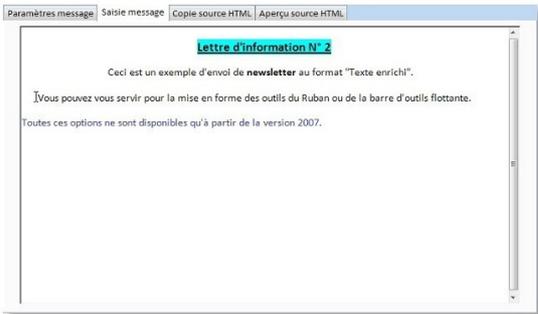
il vous permet de choisir un fichier à mettre en pièce jointe.

**5.2.1.2. Saisie du message**

Vous pouvez soit saisir un message directement avec les mises en forme offertes par le texte enrichi, soit saisir ou copier les sources d'une page HTML éditées dans un logiciel spécialisé.

**5.2.1.2.1. Saisie directe**

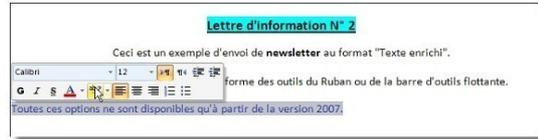
Dans cet onglet une zone de saisie vous permet de rédiger le corps de votre message.



Pour la mise en forme vous pouvez utiliser les menus appropriés dans le ruban :

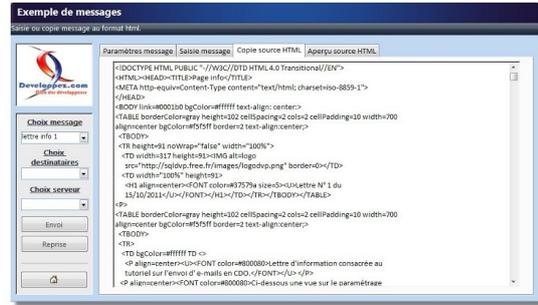


Vous pouvez également vous servir de la barre d'outils flottante :

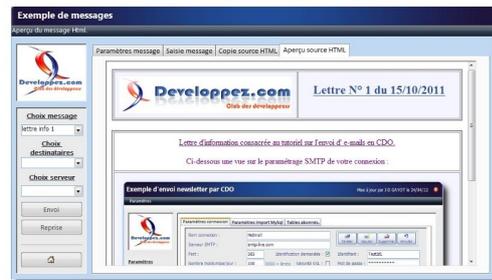


**5.2.1.2.2. Copie sources HTML**

Dans cet onglet vous pourrez saisir (si vous en avez les compétences) ou recopier les sources d'une page HTML.



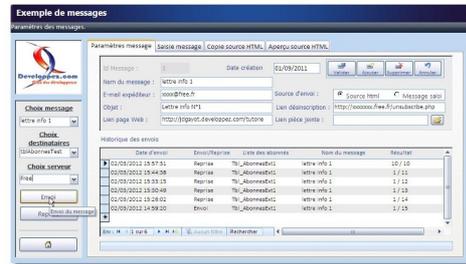
Vous pourrez visualiser la cohérence de cette source dans l'onglet "Aperçu source HTML".



**5.2.2. Envoi des messages**

Une fois le message rédigé, il reste à l'envoyer. Il faut obligatoirement :

- sélectionner un message ;
- choisir une liste de destinataires ;
- choisir un serveur d'envoi ;
- cliquer sur "Envoi".

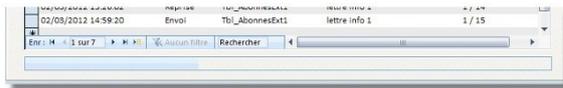


Vous devrez répondre à un message de confirmation.



L'opération d'envoi va se dérouler.

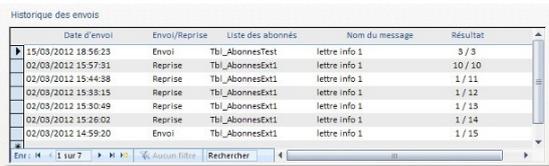
Une barre de progression en bas du formulaire vous indiquera l'avancement.



Vous serez averti à la fin de l'opération.



Une ligne sera ajoutée dans l'historique.



Le destinataire recevra suivant le cas :



ou bien :



Il y a bien en entête le lien HTML et en pied de message le lien de désinscription.

### 5.2.3. Reprise

Il peut arriver qu'il y ait un problème pendant l'envoi ou que vous ayez atteint le nombre maximum de messages défini dans les paramètres SMTP.

Dans votre historique, vous aurez par exemple 90/110. Cela voudra dire que vingt messages n'ont pas fait l'objet d'un envoi.

Dans ce cas il faut de nouveau sélectionner le message,

ainsi qu'un serveur et la liste des destinataires concernée. Puis cliquez sur "Reprise". L'envoi va reprendre là où il s'était arrêté.

Attention : dans ce type d'envoi, le résultat se fait sur les messages envoyés correctement.

Il ne préjuge pas de ceux qui seront reçus, bloqués par des firewalls et autres antispams ou surtout effectivement lus.

## 6. Développement

Comme écrit dans l'introduction, cette application est surtout destinée à aider les développeurs débutants.

Une petite application comme celle-ci permet de mettre en œuvre des problématiques variées.

Sont abordés en effet entre autres :

- manipulations sur les tables (création, copie, renommage, mise à jour de champs, parcours recordset) ;
- IHM ([Lien 28](#)) avec l'alimentation de zones de liste, le double affichage dans des sous-formulaires, ou encore une barre de progression... ;
- importation de données externes dans une table ;
- envoi d'e-mails en CDO.

Nous allons consacrer ce chapitre à ce dernier point.

### 6.1. Envoi d'e-mails en CDO

#### 6.1.1. Présentation

**CDO** ou "**Collaboration Data Objects**" est une bibliothèque permettant d'envoyer des mails sans disposer d'une messagerie sur son PC.

Différentes versions ont vu le jour :

- **CDO 1.21** ("Collaboration Data Objects") est (était) installé avec Exchange/Outlook ;
- **CDONTS** ("Collaboration Data Objects pour Windows NT Server") est la version CDO de Windows NT et 98 ;
- **CDOSYS** ("Collaboration Data Objects pour Windows 2000") a remplacé CDONTS à partir de Windows 2000/XP ;
- **CDOEX** ("CDO for Exchange Server") s'appuie sur CDOSYS et est installé avec Exchange (n'est apparemment plus installé avec la version 2010).

Si CDOEX est installé, il remplace CDOSYS :

À partir de Office 2010, CDO n'est plus inclus dans le pack et ne fonctionnera pas avec les versions Office 64 bits.

Vous pourrez donc l'utiliser sans problème de Office 2000 à 2007.

Pour les versions ayant été mises à jour vers 2010 en 32 bits, les tests que nous avons effectués se sont révélés positifs.

#### 6.1.2. Options d'envoi

Dans un premier temps, il vous faut connaître les paramètres SMTP nécessaires.

Si vous avez une messagerie installée, il vous suffit d'aller dans les paramètres d'un de vos comptes :



Le principe en est simple. Il faut créer l'objet CDO puis lui assigner plusieurs paramètres d'envoi obligatoires ou optionnels.

### 6.1.2.1. Envoi d'un message simple

Voici l'exemple d'un code utilisé pour envoyer un message tout simple.

#### envoi CDO

```
Option Compare Database
Option Explicit

Private Sub envoiCdo()
    On Error GoTo Error_send
    Dim oCdo As Object

    Set oCDO = CreateObject("CDO.Message")

    With oCDO
        With .Configuration.Fields
            .Item("http://schemas.microsoft.com/cdo/configuration/sendusing") = 2
            'ou CdoSendUsingPort : utilisation réseau
            .Item("http://schemas.microsoft.com/cdo/configuration/smtpserver") = "smtp.free.fr"
            'nom ou IP du serveur SMTP
            .Item("http://schemas.microsoft.com/cdo/configuration/smtpserverport") = "25"
            'port utilisé
            .Update

            End With
            .Subject = "envoi exemple"
            ' objet du message
            .From = "expediteur@free.fr"
            ' adresse de l'expéditeur
            .To = "destinataire@free.fr"
            ' adresse du destinataire
            .TextBody = "Ceci est un message de test."
            ' corps du message en format texte brut
            .Send

        End With

    End With

Fin:
    Set oCdo = Nothing
    Exit Sub

Error_send:
    MsgBox "Erreur d'envoi " & Err.Number & " " & Err.Description
    Resume Fin

End Sub
```

Voici le résultat dans la messagerie du destinataire :

#### envoi CDO

```
Option Compare Database
Option Explicit

Private Sub envoiCdo()
    On Error GoTo Error_send

    Dim oCdo As Object
    Dim strHtml As String 'variable contenu du corps de message

    ' Définit le contenu du message au format HTML
    strHtml = "<HTML><HEAD><BODY>"
    strHtml = strHtml & "<b> Ceci est un message de test au format <i><Font Color=#ff0000 > HTML </Font></i></b>"
    strHtml = strHtml & "</BODY></HEAD></HTML>"

    Set oCdo = CreateObject("CDO.Message")

    With oCdo
        With .Configuration.Fields
            .Item("http://schemas.microsoft.com/cdo/configuration/sendusing") = 2
            'ou CdoSendUsingPort : utilisation réseau
            .Item("http://schemas.microsoft.com/cdo/configuration/smtpserver") = "smtp.free.fr"
            'nom ou IP du serveur SMTP
            .Item("http://schemas.microsoft.com/cdo/configuration/smtpserverport") = "25"
            'port utilisé
            .Update

            End With
            .Subject = "envoi exemple"
            ' objet du message
            .From = "expediteur@free.fr"
            ' adresse de l'expéditeur
            .To = "destinataire@free.fr"
            ' adresse du destinataire
            .HtmlBody = strHtml
            ' corps du message HTML
            .Send

        End With

    End With

Fin:
    Set oCdo = Nothing
    Exit Sub

Error_send:
    MsgBox "Erreur d'envoi " & Err.Number & " " & Err.Description
    Resume Fin

End Sub
```

Le message reçu est déjà plus sympathique :



### 6.1.2.3. Envoi en Cc ou Bcc

Vous pouvez également envoyer vos messages avec les options Cc (Copie carbone) ou Bcc (Copie carbone aveugle).

#### envoi CDO

Option Compare Database

Option Explicit

```
Private Sub envoiCdo()
    On Error GoTo Error_send

    Dim oCdo As Object
    Dim strHtml As String 'variable contenu du
    corps de message

    ' Définit le contenu du message au format
    HTML
    strHtml = "<HTML><HEAD><BODY>"
    strHtml = strHtml & "<b> Ceci est un message
    de test au format <i><Font Color=#ff0000 > HTML
    </Font></i></b>"
    strHtml = strHtml & "</BODY></HEAD></HTML>"

    Set oCdo = CreateObject("CDO.Message")
    With oCdo
        With .Configuration.Fields
            .Item("http://schemas.microsoft.com/c
            do/configuration/sendusing") = 2
            'ou CdoSendUsingPort : utilisation réseau
            .Item("http://schemas.microsoft.com/c
            do/configuration/smtpserver") = "smtp.free.fr"
            'nom ou IP du serveur SMTP
```

```
.Item("http://schemas.microsoft.com/c
do/configuration/smtpserverport") = "25"
'port utilisé
.Update
End With

.Subject = "envoi exemple"
' objet du message
.From = "expediteur@free.fr"
' adresse de l'expéditeur
.To = "destinataire@free.fr"
' adresse du destinataire
.HtmlBody = strHtml
' corps du message HTML
.Bcc = "yyyy@hotmail.fr"
' adresse destinataire en copie carbone cachée
.Cc = "xxxx@orange.fr"
' adresse destinataire en copie carbone
.Send
End With

Fin:
Set oCdo = Nothing
Exit Sub

Error_send:
MsgBox "Erreur d'envoi " & Err.Number & " "
& Err.Description
Resume Fin

End Sub
```



Retrouvez la suite de l'article de Jean-Damien Gayot en ligne : [Lien 29](#)

### Manipulation des callbacks avec jQuery

Les « callbacks », ou en français « fonctions de rappel », constituent un concept fondamental du langage JavaScript, c'est pourquoi l'équipe derrière la librairie jQuery a décidé de leur dédier spécifiquement tout un pan de leur API.

Cet article se propose de présenter les aspects fondamentaux de la manipulation de callbacks avec jQuery.

#### 1. Rappel sur la notion de callback

##### 1.1. Qu'est-ce qu'un callback ?

JavaScript tel qu'il est implémenté actuellement par les différents navigateurs est *monothread*, c'est-à-dire que le moteur d'exécution du code ne peut effectuer qu'une seule opération à la fois (contre plusieurs en parallèle s'il était *multithread*). Cela constitue un problème car tant qu'un code JavaScript s'exécute l'interface Web reste bloquée, ce qui provoque une dégradation de l'expérience utilisateur.

Pour contourner ce problème, on utilise généralement les capacités du langage JavaScript à effectuer des opérations asynchrones, c'est-à-dire à libérer le thread de l'interface utilisateur en attendant que certaines conditions soient remplies pour continuer l'exécution du code.

Il est très simple de simuler un appel asynchrone en JavaScript en utilisant les fonctions `setTimeout()` ou `setInterval()`. Celles-ci permettent de différer l'exécution d'une fonction selon un intervalle de temps laissé libre au développeur, la différence étant que dans le premier cas, l'exécution n'aura lieu qu'une seule fois et dans le second, l'exécution sera répétée à intervalle régulier.

```
// Cette fonction sera exécutée dans 1000
millisecondes
setTimeout(
  function () {
    console.log("Exécution différée d'une
seconde.");
  },
  1000
);
```

L'avantage de ces deux fonctions est que pendant la durée d'attente, l'interface utilisateur peut réagir aux interactions de celui-ci. Pourtant, en tâche de fond, le chronomètre tourne toujours.

Dans cet exemple, la fonction anonyme définie pour s'exécuter lorsque le temps se sera écoulé est un exemple typique de callback : elle ne s'exécute qu'en réaction à la réalisation de certaines conditions (ici en l'occurrence, à la fin du décompte du chronomètre).

##### 1.2. Comment jQuery utilise les callbacks

La librairie jQuery fait usage des callbacks dans deux cas précis :

- dans l'API d'animation, grâce à laquelle on peut chaîner différentes animations pour que celles-ci s'exécutent les unes à la suite des autres. Dans ce cas, ces animations constituent elles-mêmes des callbacks, puisque leur exécution sera dépendante de la fin de l'animation précédente ;
- dans l'API Ajax, dont la nature est par définition asynchrone (bien qu'on puisse l'utiliser volontairement en mode synchrone). Il est en effet impossible de connaître à l'avance la durée d'une opération d'appel vers un serveur. On utilise donc des callbacks pour indiquer comment devra réagir le client lorsque le serveur aura répondu.

```
/* Exemple d'utilisation asynchrone de l'API
d'animation */
// Il y aura un décalage de 800 millisecondes
entre les animations "slideUp" et "fadeIn"
$(
  "div#element").slideUp(400).delay(800).fadeIn(400);
```

```
/* Exemple d'utilisation asynchrone de l'API Ajax
*/
// Les fonctions anonymes définies dans "success"
et "error" ne s'exécuteront qu'une fois l'appel
Ajax terminé,
// et en fonction de la réussite ou de l'échec de
celui-ci
$.ajax({
  "url":
"http://odata.netflix.com/Catalog/Titles" +
  "?$filter=substringof('indiana
jones',Name)&$format=json&$callback=callback",
  "dataType": 'jsonp',
  "type": 'GET',
  "jsonpCallback": 'callback',
  "success": function (data, textStatus, jqXHR)
{
  console.log("L'appel Ajax est une
réussite.");
},
  "error": function (jqXHR, textStatus,
errorThrown) {
  console.log("L'appel Ajax est un
échec.");
}
});
```

Notez qu'il est aussi possible dans le cas de l'API Ajax de définir non pas une seule fonction de callback, mais un tableau de celles-ci, qui seront alors exécutées à tour de rôle.

Ces deux exemples illustrent bien l'intérêt des callbacks dans l'utilisation de jQuery. Toutefois, leur omniprésence a conduit les membres de l'équipe de développement de la célèbre librairie à s'interroger sur la meilleure manière de les représenter. C'est ainsi qu'a été introduite à partir de la version 1.5 toute une API destinée à leur manipulation.

## 2. Consommer des callbacks dans jQuery

L'arrivée dans jQuery d'une API qui leur est dédiée a permis une refonte complète de l'implémentation interne des callbacks. Les API d'animation et Ajax sont les premières à en bénéficier, et les objets retournés par leurs différentes méthodes ont été augmentés pour les rendre compatibles avec la notion de callback.

Ces objets disposent maintenant d'une méthode `promise()` qui est le point d'entrée de l'ajout de callbacks. Une promise (« promesse ») est une version réduite de l'objet utilisé par jQuery pour gérer les callbacks, permettant seulement de rattacher ces derniers, sans possibilité de résoudre l'opération sous-jacente. Mais nous y reviendrons.

À partir de cette méthode `promise()`, il est possible d'empiler nos différents callbacks. Ceux-ci peuvent être de trois types :

- les callbacks en cas de réussite (« done callbacks ») ;
- les callbacks en cas d'échec (« fail callbacks ») ;
- les callbacks lors de la notification de la progression de l'opération (« progress callbacks »). Ceci constitue une nouveauté introduite dans jQuery 1.7.

Les callbacks rattachés à la promise seront déempilés et exécutés dans l'ordre. Les paramètres qui leur seront passés seront dépendants du type de l'opération ainsi que de la réussite ou de l'échec de celle-ci. Par exemple, en cas de réussite d'une requête Ajax, les paramètres passés aux callbacks seront ceux passés traditionnellement à la propriété « success » de l'API Ajax (*data*, *textStatus* et *jqXHR*).

Voyons maintenant comment effectivement rattacher des callbacks à une promise.

### 2.1. Manipuler une seule promise

Rattacher des callbacks à une promise unique se fait par le biais des méthodes `done()`, `fail()` et `progress()`.

```
$(document).ready(function () {
    var jQueryObj = $
    ("#element").slideUp().delay(1500).fadeIn();
    jQueryObj.promise()
        .done(function ()
        { console.log("L'animation est réussie."); })
        .fail(function ()
        { console.log("L'animation est ratée."); })
        .progress(function ()
        { console.log("L'animation est en cours de
        progression."); });
});
```

Dans cet exemple, `jQueryObj` est un objet jQuery tout ce

qu'il y a de plus traditionnel, tel qu'il est renvoyé par la majorité des méthodes chaînables de jQuery, notamment les méthodes d'animation. Mais cet objet jQuery dispose d'une nouvelle fonction `promise()` à partir de laquelle il est possible de rattacher des callbacks.

En l'occurrence, si vous exécutez le code, vous verrez que seul le texte « *L'animation est réussie.* » s'affiche à la fin de l'opération, car les fonctions d'animation ne notifient pas leur progression et ne sont normalement jamais en échec.

Notez que toutes ces méthodes acceptent aussi bien une seule fonction de callback qu'un tableau de celles-ci, auquel cas ces fonctions seront déempilées et exécutées à tour de rôle.

Il existe aussi une autre fonction permettant de rattacher un callback qui sera exécuté dans tous les cas, que l'opération soit une réussite ou un échec. Il s'agit de la fonction `always()`.

```
$(document).ready(function () {
    var ajaxCall = $.ajax({
        "url":
        "http://odata.netflix.com/Catalog/Titles" +
        "?$filter=substringof('indiana
        jones',Name)&$format=json&$callback=callback",
        "type": 'GET',
        "dataType": 'jsonp',
        "jsonpCallback": 'callback'
    });

    ajaxCall
        .always([
            function () { console.log("L'appel
            Ajax est terminé."); },
            function () { console.log("J'ai déjà
            dit que l'appel Ajax était terminé."); }
        ]);
});
```

Dans cet exemple, nous passons à la méthode `always()` deux fonctions de callback qui seront donc exécutées à tour de rôle, en respectant l'ordre de leur index dans le tableau.

Contrairement à un objet jQuery traditionnel, il n'est pas nécessaire d'appeler la méthode `promise()` sur un objet `jqXHR` tel que renvoyé par la méthode `jQuery.ajax()`.

Il existe un piège dans lequel il vous faut prendre garde de ne pas tomber lors de l'utilisation des callbacks avec `jQuery.ajax()`. Il existe en effet dans cette API une option « *beforeSend* » qui permet d'altérer l'objet `jqXHR` avant l'envoi de la requête (pour modifier les en-têtes HTTP par exemple), mais qui permet aussi d'annuler purement et simplement la requête si la fonction de traitement de l'option renvoie `false`.

Le piège est que lorsque l'on renvoie `false`, `jQuery.ajax()` ne renvoie pas une promise mais également `false`. Si vous utilisez l'option « *beforeSend* » et cette particularité, il vous faut donc vérifier que la requête a bien été envoyée avant d'essayer de manipuler la promise.

Ceci constitue un comportement un peu gênant qui a été rapporté à l'équipe de développement de jQuery, et qui ne sera vraisemblablement pas corrigé dans un futur proche :

## Lien 30.

Inconvénient non négligeable : il est impossible de savoir à l'avance quels seront les paramètres passés aux callbacks, puisque ceux-ci seront susceptibles de différer en fonction de la réussite ou de l'échec de l'opération. Il faudra donc en analyser soi-même la nature. La méthode `always()` se prête donc davantage à des scénarios où les paramètres et l'état final de l'opération importent peu.

Il est bien sûr possible d'obtenir facilement le même résultat que l'utilisation de la méthode `always()` en passant la même fonction aux méthodes `done()` et `fail()` :

```
var onComplete = function () {
    console.log("L'appel Ajax est terminé.");
};

ajaxCall
    .done([
        function () { console.log("L'appel Ajax est réussi."); },
        onComplete
    ])
    .fail([
        function () { console.log("L'appel Ajax est raté."); },
        onComplete
    ]);
```

Il est aussi possible d'utiliser la méthode `then()`, purement utilitaire, qui permet d'attacher en une seule fois des callbacks de réussite, d'échec et de notification de la progression :

```
ajaxCall.then(
    function () { console.log("L'appel Ajax est réussi."); }, // Callback de réussite
    function () { console.log("L'appel Ajax est raté."); }, // Callback d'échec
    function () { console.log("L'appel Ajax est en cours."); } // Callback de progression
);
```

Tout ce que nous avons vu jusqu'à maintenant, il était déjà possible de l'effectuer sans l'API de callback. Ce qu'apporte essentiellement cette dernière, c'est la possibilité de traiter les callbacks de manière plus globalisée et en particulier, de consolider plusieurs promesses pour n'exécuter les callbacks que lorsqu'elles sont toutes résolues. C'est le rôle de la méthode `jQuery.when()`.

### 2.2. Consolider plusieurs promesses avec `jQuery.when()`

Grâce à `jQuery.when()`, nous sommes maintenant en mesure de regrouper plusieurs promesses afin d'y rattacher des callbacks qui ne seront exécutés que lorsque l'ensemble des opérations sous-jacentes seront résolues.

```
// La déclaration des variables jQueryObj et ajaxCall est omise par souci de brièveté
$.when(jQueryObj, ajaxCall)
    .done(function (animParameters,
ajaxParameters) {
    console.log("Toutes les opérations se sont terminées normalement.");
});
```

`jQuery.when()` retourne une promesse qui regroupe toutes les autres promesses passées en paramètre. Nous pouvons ensuite rattacher des callbacks à cette *master promise* de la même manière qu'expliqué précédemment.

Dans cet exemple, nous regroupons l'opération d'animation et l'opération d'appel Ajax. Le callback de réussite que nous attachons à la nouvelle promesse sera exécuté dès lors que ces deux opérations auront été successivement résolues. Par contre, si une seule de ces opérations devait tomber en échec, la promesse serait instantanément rejetée.

Autre élément important : les paramètres de résolution sont regroupés dans des tableaux et passés aux callbacks dans le même ordre que les promesses sont passées à `jQuery.when()`. Ainsi dans notre exemple, `ajaxParameters` sera un tableau contenant tous les paramètres de résolution d'une requête `jQuery.ajax()`, à savoir les données retournées (`ajaxParameters[0]`), le statut de la requête (`ajaxParameters[1]`) et l'objet `jqXHR` (`ajaxParameters[2]`).

## 3. Comment fonctionne l'API de callbacks de `jQuery`

L'API de manipulation de callbacks de `jQuery` fonctionne en fait autour de deux API distinctes, l'API `jQuery.Callbacks`, et l'API `jQuery.Deferred`.

### 3.1. L'API `jQuery.Callbacks`

L'API `jQuery.Callbacks` permet de créer des listes de fonctions dont l'exécution pourra ensuite être déclenchée au moment opportun.

#### 3.1.1. La manipulation des listes

Une liste de fonctions se crée en appelant le constructeur `jQuery.Callbacks()`. Celui-ci prend optionnellement une liste de « flags » sur laquelle nous reviendrons bientôt.

Les méthodes `add()`, `remove()` et `empty()` permettent ensuite facilement d'ajouter et supprimer des fonctions à la liste, ou de vider celle-ci.

Notez que contrairement à ce qu'indique la documentation de `jQuery`, les fonctions de l'API `jQuery.Callbacks` sont parfaitement chaînables.

En outre, les méthodes d'ajout et de suppression peuvent prendre en paramètre une seule fonction ou un tableau de celles-ci.

```
var externalFunction = function () {
    console.log("Fonction nommée.");
};

var listOfCallbacks = $.Callbacks()
    // On ajoute une fonction anonyme et la
    // fonction nommée "externalFunction" à la liste
    .add([
        function () { console.log("Fonction
anonyme."); },
        externalFunction
    ])
    // On retire la fonction nommée
    // "externalFunction" de la liste
    .remove(externalFunction);
```

Grâce à la fonction `has()` il est aussi possible de déterminer si une fonction est présente ou non dans une liste.

```
// hasFunction contiendra "false" car nous avons
retiré la fonction nommée "externalFunction"
auparavant
var hasFunction =
listOfCallbacks.has(externalFunction);
```

Une fois notre liste constituée, il est possible d'en exécuter les fonctions à loisir grâce aux méthodes `fire()` et `fireWith()`.

```
var listOfCallbacks = $.Callbacks()
.add(
function (text) {
console.log("Votre texte : " + text);
console.dir(this);
}
)
// Affichera :
// "Votre texte : toto"
// Le dump Firebug de l'objet listOfCallbacks
.fire("toto")
// "Votre texte : titi"
// Le dump Firebug de l'objet window
.fireWith(window, [ "titi" ]);
```

La différence est que `fire()` prend simplement en paramètres les arguments qui seront transmis aux différents callbacks, tandis que `fireWith()` permet de modifier le contexte d'exécution de ceux-ci, de la même manière qu'on le ferait avec la fonction JavaScript `apply()` (que jQuery appelle d'ailleurs sous le manteau), c'est-à-dire en passant un nouveau contexte en guise de premier paramètre et un tableau d'arguments en guise de second paramètre.

Accessoirement la méthode `fired()` permet de savoir si les fonctions de la liste ont été exécutées au moins une fois. Notez que le fait de modifier le contenu de la liste ou même de vider celle-ci entretemps n'y change rien.

```
// wasFired contiendra "true" car nous avons
exécuté la liste de fonctions au moins une fois,
et ce
// bien que nous ayons vidé celle-ci entretemps
listOfCallbacks.empty();
var wasFired = listOfCallbacks.fired();
```

### 3.1.2. Le système de flags

Au moment de créer une liste de fonctions avec le constructeur de `jQuery.Callbacks()`, il est possible de passer à celui-ci une liste de flags sous forme de chaînes de caractères. Ces flags sont de simples mots-clés qui vont permettre d'altérer le comportement de la liste. Leur usage est cumulatif.

Ces mots-clés sont actuellement au nombre de quatre :

- « **once** » permet de limiter l'exécution de la liste à une seule fois. Les exécutions suivantes seront sans effet ;
- « **unique** » permet de s'assurer qu'un callback ne pourra être ajouté qu'une seule fois à la liste. Ajouter un callback à une liste qui le contient déjà sera sans effet et sa position dans la file d'exécution n'en sera pas altérée ;

- « **memory** » permet de conserver en mémoire les valeurs passées en paramètre lors de la première exécution de la liste. L'ajout ultérieur de nouveaux callbacks provoquera l'exécution immédiate de ceux-ci avec les valeurs en mémoire. Il est toujours possible d'exécuter à nouveau la liste de fonctions, et donc de modifier la liste des paramètres en mémoire ;
- « **stopOnFalse** » permet d'indiquer que l'on souhaite que l'exécution de la liste de fonctions s'interrompe si l'une de celles-ci retourne false.

```
/* Exemple d'utilisation du mot-clé "once" */
// Ne s'affichera que "Votre texte est : toto",
les exécutions suivantes étant ignorées
$.Callbacks("once")
.add(function (text) {
console.log("Votre texte est : " + text);
})
.fire("toto")
.fire("titi")
.fire("tutu");
```

```
/* Exemple d'utilisation du mot-clé "unique" */
var externalFunction = function (text) {
console.log("Votre texte est : " + text);
};
// Affichera "Votre texte est : toto" une seule
fois car les ajouts multiples d'une même fonction
sont ignorés
$.Callbacks("unique")
.add(externalFunction)
.add(externalFunction)
.fire("toto");
```

```
/* Exemple d'utilisation du mot-clé "memory" */
// Affichera "Votre texte est : toto" au moment
de l'appel à fire()
// Puis "Vous avez saisi : toto" tout de suite
après l'ajout du nouveau callback
$.Callbacks("memory")
.add(function (text) {
console.log("Votre texte est : " + text);
})
.fire("toto")
.add(function (text) {
console.log("Vous avez saisi : " + text);
});
```

```
/* Exemple d'utilisation du mot-clé
"stopOnFalse" */
// N'affichera pas "Votre texte est : toto" car
le premier callback retourne "false",
interrompant ainsi
// l'exécution du reste de la liste
$.Callbacks("stopOnFalse")
.add([
function () {
console.log("J'interromps l'exécution
de la liste");
return false;
},
function (text) {
console.log("Votre texte est : " +
text);
}
])
.fire("toto");
```

Comme on le voit, l'éventail des possibilités offertes par ces mots-clefs permet déjà des usages assez variés. Par ailleurs, comme déjà expliqué, il est possible de cumuler l'effet de ces différents mots-clefs, afin d'affiner encore davantage la manière dont pourra être manipulée une liste.

```
/* Exemple d'utilisation cumulée des mots-clefs
"once" et "memory" */
// Affichera "Votre texte est : toto"
// puis "Vous avez saisi : toto" dès l'ajout du
second callback
$.Callbacks("once memory")
  .add(function (text) {
    console.log("Votre texte est : " + text);
  })
  .fire("toto")
  .fire("titi")
  .add(function (text) {
    console.log("Vous avez saisi : " + text);
  });
```

Dans cet exemple, le deuxième appel à fire() est inhibé par le mot-clef « once », mais l'utilisation du mot-clef « memory » permet malgré tout d'exécuter instantanément les callbacks ajoutés ultérieurement. Par contre, le cumul de ces deux mots-clefs a aussi pour effet de geler une bonne fois pour toutes les valeurs des paramètres enregistrées en mémoire. Ainsi, comme on le voit ici, c'est bien la valeur « toto » qui est retenue lors de l'ajout du second callback, la seconde exécution avec la valeur « titi » ayant été totalement ignorée.

Vous aurez remarqué que dans l'API jQuery.Callbacks, il n'est pas question de « promise » ou de différents types de callbacks de réussite ou d'échec. En effet, cette API se contente de jouer un rôle bas niveau, en proposant le strict minimum pour gérer des listes de fonctions. L'API haut niveau n'est autre que jQuery.Deferred.

### 3.2. L'API jQuery.Deferred

L'API jQuery.Deferred repose sur l'API jQuery.Callbacks. C'est ici que nous allons retrouver notre notion de « promise », ainsi que nos différents types de callbacks. Pour gérer ces derniers, l'API jQuery.Deferred utilise des objets jQuery.Callbacks avec les flags « once » et « memory » (sauf pour la notification de progression, pour laquelle il n'utilise que « memory »).

jQuery.Deferred est l'API de choix pour gérer des listes de callbacks sur des opérations asynchrones et c'est justement elle qu'utilise jQuery dans les API d'animation et Ajax. Lorsque l'on manipule ces dernières, on n'a accès qu'à des promesses, ne permettant que de rattacher des callbacks. Mais nous allons voir maintenant tout l'éventail de possibilités que permet jQuery.Deferred.

#### 3.2.1. Manipulation d'un objet jQuery.Deferred

Comme nous l'avons déjà dit, une promise ne permet que de rattacher des callbacks à un objet qui a seul le contrôle de l'exécution de ceux-ci. Ce fameux objet est en fait l'objet jQuery.Deferred, qui va donc décider du moment où se fera cette exécution et de la manière dont elle se fera.

Lorsque l'on manipule un objet jQuery.Deferred, le principe est donc de n'exposer que sa promise afin de

rester maître du déroulement asynchrone des opérations.

```
/* Exemple d'utilisation d'un objet
jQuery.Deferred */
// On appelle une fonction qui renvoie une
promise
operation().done(function (magicNumber) {
  console.log("Le nombre magique est " +
  magicNumber);
});

function operation() {
  var dfd = $.Deferred();

  // L'objet jQuery.Deferred ne sera résolu
  qu'au bout de 5 secondes
  setTimeout(function () {
    dfd.resolve(42);
  }, 5000);

  return dfd.promise();
}
```

On voit dans cet exemple que la fonction operation() agit comme une boîte noire. Le code appelant ne dispose que d'une promise, et ne sait pas quand et avec quels paramètres celle-ci sera résolue. Cette promise est renvoyée grâce à la méthode promise() de l'objet jQuery.Deferred, dont le constructeur peut prendre optionnellement deux paramètres : une chaîne de caractères indiquant à quelle queue la promise est liée (par exemple la queue « fx » pour les méthodes de l'API d'animation) et un objet auquel on souhaite rattacher la promise, comme le fait l'API Ajax avec l'objet jqXHR.

Notez que vous pouvez tout à fait choisir d'exposer l'intégralité de l'objet jQuery.Deferred, déléguant ainsi le choix de l'exécution des callbacks, mais ce n'est pas vraiment le principe souhaité d'utilisation, et l'API jQuery.Callbacks convient mieux dans ce cas.

Il existe trois méthodes permettant de notifier l'état d'avancement d'un objet jQuery.Deferred :

- **resolve()** qui permet de résoudre l'objet ;
- **reject()** qui permet de rejeter l'objet ;
- **notify()** qui permet d'indiquer l'avancement de l'opération. Ceci constitue une nouveauté introduite dans jQuery 1.7.

```
/* Exemple d'utilisation des fonctions de
résolution de l'objet jQuery.Deferred */
operation().then(
  // Réussite
  function (response) {
    console.log("L'opération est terminée
    avec succès et la réponse est : " + response);
  },
  // Échec
  function (error) {
    console.log("L'opération a raté à cause
    de l'erreur : " + error);
  },
  // Avancement
  function (progress) {
    console.log("L'opération en est
    actuellement à l'étape : " + progress);
  }
);
```

```
function operation() {
    var dfd = $.Deferred();
    var zeroOrOne = Math.round(Math.random());

    setTimeout(function () {
        dfd.notify("Je m'échauffe");
    }, 1000);

    setTimeout(function () {
        dfd.notify("Prêt à l'action");
    }, 2000);

    setTimeout(function () {
        if (zeroOrOne === 1)
            dfd.resolve("Le chiffre un est tombé");
        else
            dfd.reject("Le chiffre zéro est tombé");
    }, 3000);

    return dfd.promise();
}
```



Dans cet exemple, nous utilisons une nouvelle fois la fonction `setTimeout()` native du langage JavaScript pour simuler une opération asynchrone. Ainsi, au bout d'une seconde la fonction `operation()` va notifier une première fois les callbacks de progression attachés à sa promesse pour leur indiquer qu'elle « s'échauffe », puis au bout d'une deuxième seconde, indiquer qu'elle est « prête à l'action », et enfin au bout d'une troisième seconde, résoudre ou rejeter l'objet `jQuery.Deferred` en fonction d'une valeur entre 0 et 1 tirée au sort aléatoirement avec la fonction native `Math.random()`. La capture d'écran montre le résultat de ces résolutions successives de callbacks dans la console de Firefox ([Lien 31](#)).

Notez que ces trois méthodes existent aussi dans une version alternative permettant d'altérer le contexte d'exécution des callbacks qui s'y rattacheront : `resolveWith()`, `rejectWith()` et `notifyWith()`. Comme pour la fonction `fireWith()` de l'API `jQuery.Callbacks`, celles-ci utilisent sous le manteau la fonction JavaScript `apply()`.

Celui qui contrôle l'objet `jQuery.Deferred` est seul maître de son état. Au moment de la création d'un objet, celui-ci sera toujours *pending* (« en attente »), c'est-à-dire qu'il n'y a encore eu ni résolution ni rejet. Il est possible de contrôler l'état d'un objet `jQuery.Deferred` grâce à la méthode `state()` qui renvoie une chaîne de caractères pouvant prendre les valeurs *pending*, *resolved* ou *rejected*. La promesse a également accès à cette méthode et à ces informations.

Nous avons déjà dit que pour gérer les callbacks, `jQuery.Deferred` reposait sur l'utilisation de `jQuery.Callbacks` avec les flags « once » et « memory », et par ailleurs, nous avons étudié plus tôt le résultat de

l'utilisation conjointe de ceux-ci. Nous pouvons donc en déduire deux affirmations :

- une fois qu'un objet `jQuery.Deferred` a été transitionné dans l'état *resolved* ou *rejected*, son état ne peut plus changer, et les callbacks ajoutés ultérieurement aux méthodes `done()` ou `fail()` de sa promesse seront exécutés immédiatement. Par contre, la notification de progression avec `notify()` peut être effectuée plusieurs fois tant que l'objet est dans l'état *pending* ;
- les paramètres utilisés lors de la résolution ou du rejet d'un objet `jQuery.Deferred` ne peuvent plus être modifiés après coup. Ceux qui sont passés au moment de la transition de l'état de l'objet `jQuery.Deferred` seront conservés en mémoire et passés en paramètre de tout callback qui pourrait être attaché ultérieurement. Là encore, la notification de progression fait exception car il est possible de modifier les paramètres d'exécution des callbacks qui s'y rattachent.

```
/* Exemple de comportement de jQuery.Deferred */
// Affichera "L'opération est ratée."
operation().then(
    // Réussite
    function (response) {
        console.log("L'opération est réussie.");
    },
    // Échec
    function (error) {
        console.log("L'opération est ratée.");
    }
);

function operation() {
    var dfd = $.Deferred();

    // Exécution immédiate de l'objet
    jQuery.Deferred
    dfd.reject();

    // Ne sert à rien car l'objet a déjà été
    rejeté
    dfd.resolve();

    return dfd.promise();
}
```

Il y a deux choses à noter dans ce dernier exemple. Tout d'abord, l'objet `jQuery.Deferred` est rejeté tout de suite après sa création, et les callbacks qui sont attachés à sa promesse, même si cette opération n'a lieu qu'après le rejet, sont donc exécutés instantanément. Par ailleurs, la résolution qui a lieu dans un second temps est totalement ignorée, ce qui signifie que l'état de l'objet `jQuery.Deferred` n'est pas altéré et que les callbacks de résolution attachés à sa promesse ne seront pas exécutés.

Ce sont toutes ces règles respectant le design CommonJS Promises/A ([Lien 32](#)) qui permettent d'avoir une API `jQuery.Deferred` à la fois souple et robuste.

### 3.2.2. Filtrer et chaîner les callbacks avec la fonction `pipe()`

Une autre méthode utilitaire de l'API `jQuery.Deferred` qui mérite que l'on s'y attarde est la méthode `pipe()`, qui

permet de filtrer les paramètres passés aux callbacks lors de la résolution, le rejet, ou la notification de la promesse. Cette méthode est accessible aussi bien par l'objet maître `jQuery.Deferred` que par ses promesses.

```
/* Exemple de filtrage avec pipe() */
// Affichera "L'OPÉRATION EST EN COURS."
// Affichera "Est-ce que le tableau est vide : false"
operation()
    .done(function (arrayIsEmpty) {
        console.log("Est-ce que le tableau est vide : " + arrayIsEmpty);
    })
    .progress(function (comment) {
        console.log(comment);
    });

function operation() {
    var dfd = $.Deferred();

    setTimeout(function () {
        dfd.notify("L'opération est en cours.");
    }, 1000);

    setTimeout(function () {
        dfd.resolve([ "toto" ]);
    }, 2000);

    return dfd.pipe(
        // Filtrage des paramètres de résolution
        function (array) {
            return array.length === 0;
        },
        // Pas de filtrage sur les paramètres de
rejet
        null,
        // Filtrage des paramètres de
notification de progression
        function (comment) {
            return comment.toUpperCase();
        }
    );
}
```

Dans cet exemple la méthode `pipe()` filtre les paramètres des callbacks de résolution et de progression. En passant « null » à `pipe()` en guise de fonction filtrant les paramètres de rejet, nous indiquons que nous voulons que ceux-ci ne soient pas altérés.

Le paramètre de la notification de progression est une chaîne de caractères, qui grâce à la fonction `pipe()` sera passée en majuscules avant d'être passée aux callbacks rattachés à la promesse. Le paramètre de la résolution est théoriquement un tableau, mais nous l'interceptons pour renvoyer à la place un booléen indiquant si celui-ci est vide.

Notez bien encore une fois que ces filtrages peuvent aussi bien être appliqués au niveau de l'objet `jQuery.Deferred` lui-même qu'au niveau de ses promesses, et qu'en outre, la méthode `pipe()` renvoie elle-même une nouvelle promesse, ce que démontre là aussi notre exemple.

Mais la méthode `pipe()` a une autre utilité. Elle permet en effet de chaîner les appels asynchrones, afin de n'obtenir que le résultat final.

```
/* Exemple de chaînage avec pipe() */
// Affichera "Les données ont été chargées."
// Affichera "L'opération a réussi."
operation()
    .done(function () {
        console.log("L'opération a réussi.");
    })
    .progress(function (status) {
        console.log(status);
    });

function operation() {
    var dfd = $.Deferred();

    var promiseAjax = $.ajax({
        "url":
"http://odata.netflix.com/Catalog/Titles" +
        "?
$filter=substringof('macross',Name)&$format=json&
$callback=callback",
        "dataType": 'jsonp',
        "type": 'GET',
        "jsonpCallback": 'callback'
    });

    promiseAjax.done(function () {
        dfd.notify("Les données ont été
chargées.");
    });

    var promiseAnimation =
promiseAjax.pipe(function (data) {
        var list = $("ul#list");
        $.each(data.d.results, function (i, item)
        {
            $
("<div>").text(item.Name).appendTo(list);
        });
        dfd.resolve();
        return
list.delay(1000).slideDown().promise();
    });

    return $.when(dfd, promiseAnimation);
}
```

Il se passe beaucoup de choses dans cet exemple. Nous chargeons des données grâce à un appel Ajax asynchrone, mais puisque nous ne contrôlons pas celui-ci, nous créons un autre objet `jQuery.Deferred` pour notifier le code appelant du moment où les données ont été effectivement chargées. Par ailleurs, nous chaînons grâce à la méthode `pipe()` la promesse de cet appel Ajax avec une routine qui va charger le résultat dans une liste à puces et animer l'affichage de celle-ci. Normalement la méthode `pipe()` retourne une nouvelle promesse prenant en compte le filtrage que nous avons appliqué. Mais ici nous indiquons que nous souhaitons retourner la promesse renvoyée par la file d'animation.

Finalement, nous renvoyons au code appelant notre objet `jQuery.Deferred` consolidé avec la promesse de la file d'animation. Cette astuce nous permet à la fois de notifier le code appelant car nous avons la main sur l'objet maître `jQuery.Deferred`, mais aussi de ne résoudre l'ensemble de l'opération que lorsque l'animation d'affichage est terminée, alors qu'à ce stade l'objet maître a lui déjà été résolu.

#### **4. Conclusion**

Le monde du Web est en pleine effervescence, car un nouveau modèle d'application Web commence à prendre beaucoup de place sur le devant de la scène : les *single page applications*, dans lesquelles une seule page web gère l'ensemble d'un site. Ce modèle repose sur des interactions entre la page et le serveur sous forme d'appels Ajax (ou bientôt WebSockets), mais aussi sur de nombreuses animations pour remplacer la navigation de page en page. Cela implique que les fonctions asynchrones

vont prendre une place centrale dans l'éco

#### **4.1. Pour aller plus loin**

La documentation de jQuery sur les API Callbacks et Deferred :

- jQuery.Callbacks : [Lien 33](#) ;
- jQuery.Deferred : [Lien 34](#).

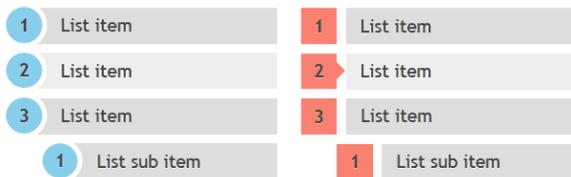
*Retrouvez l'article de François Guillot en ligne : [Lien 35](#).*

### Ajoutez du style à vos listes ordonnées

Ajouter des styles à des listes ordonnées a longtemps été une tâche difficile et piégeuse, je ne suis pas le seul à le penser. Pour donner un style aux nombres, vous devez d'abord supprimer le style par défaut du navigateur puis ajuster le code des éléments de la liste afin de leur donner le style approprié.

Dans cet article, je vais vous montrer comment utiliser CSS3 pour améliorer la présentation de vos listes ordonnées en utilisant une approche sémantique.

#### 1. L'idée



Lorsque j'ai lu pour la première fois l'article ([Lien 36](#)) de Roger Johansson sur les styles des listes ordonnées, je dois avouer que j'ai tout de suite aimé sa technique. Je vais essayer de l'utiliser et de l'améliorer un petit peu pour vous montrer différentes façons de mettre en forme vos listes ordonnées.

Vous pouvez déjà voir un exemple en ligne : [Lien 37](#).

#### 2. Le HTML

Le balisage utilisé est tout ce qu'il y a de plus classique :

```
<ol class="rounded-list">
  <li><a href="">List item</a></li>
  <li><a href="">List item</a></li>
  <li><a href="">List item</a>
    <ol>
      <li><a href="">List sub item</a></li>
      <li><a href="">List sub item</a></li>
      <li><a href="">List sub item</a></li>
    </ol>
  </li>
  <li><a href="">List item</a></li>
  <li><a href="">List item</a></li>
</ol>
```

#### 3. Le CSS

Essayons d'expliquer simplement comment cela fonctionne.

Nous allons utiliser le contenu généré, la numérotation automatique et les listes ([Lien 38](#)). Il s'agit essentiellement d'utiliser deux propriétés CSS2.1 : counter-reset (initialise un compteur) et counter-increment (le nom est explicite, cela incrémente le compteur précédent). Comme vous le verrez dans le code ci-dessous, counter-increment sera utilisé en relation avec du contenu généré CSS (pseudoéléments : [Lien 39](#)).

```
ol{
  counter-reset: li; /* Initialise le
```

```
compteur */
  list-style: none; /* Supprime la
numérotation par défaut */
  *list-style: decimal; /* Restitue la
numérotation par défaut pour IE6/7 */
  font: 15px 'trebuchet MS', 'lucida sans';
  padding: 0;
  margin-bottom: 4em;
  text-shadow: 0 1px 0
  rgba(255,255,255,.5);
}
ol ol{
  margin: 0 0 0 2em; /* Ajoute une marge à
gauche pour les listes imbriquées */
}
```

#### 3.1. Numérotation arrondie



```
.rounded-list a{
  position: relative;
  display: block;
  padding: .4em .4em .4em 2em;
  *padding: .4em;
  margin: .5em 0;
  background: #ddd;
  color: #444;
  text-decoration: none;
  border-radius: .3em;
  transition: all .3s ease-out;
}
.rounded-list a:hover{
  background: #eee;
}
.rounded-list a:hover:before{
  transform: rotate(360deg);
}
.rounded-list a:before{
  content: counter(li);
  counter-increment: li;
  position: absolute;
  left: -1.3em;
  top: 50%;
```

```

margin-top: -1.3em;
background: #87ceeb;
height: 2em;
width: 2em;
line-height: 2em;
border: .3em solid #fff;
text-align: center;
font-weight: bold;
border-radius: 2em;
transition: all .3s ease-out;
}

```

### 3.2. Numérotation rectangulaire



```

.rectangle-list a{
  position: relative;
  display: block;
  padding: .4em .4em .4em .8em;
  *padding: .4em;
  margin: .5em 0 .5em 2.5em;
  background: #ddd;
  color: #444;
  text-decoration: none;
  transition: all .3s ease-out;
}

.rectangle-list a:hover{
  background: #eee;
}

.rectangle-list a:before{
  content: counter(li);
  counter-increment: li;
  position: absolute;
  left: -2.5em;
  top: 50%;
  margin-top: -1em;
  background: #fa8072;
  height: 2em;
  width: 2em;
  line-height: 2em;
  text-align: center;
  font-weight: bold;
}

.rectangle-list a:after{
  position: absolute;
  content: '';
  border: .5em solid transparent;
  left: -1em;
  top: 50%;
  margin-top: -.5em;
  transition: all .3s ease-out;
}

.rectangle-list a:hover:after{
  left: -.5em;
  border-left-color: #fa8072;
}

```

### 3.3. Numérotation cerclée



```

.circle-list li{
  padding: 2.5em;
  border-bottom: 1px dashed #ccc;
}

.circle-list h2{
  position: relative;
  margin: 0;
}

.circle-list p{
  margin: 0;
}

.circle-list h2:before{
  content: counter(li);
  counter-increment: li;
  position: absolute;
  z-index: -1;
  left: -1.3em;
  top: -.8em;
  background: #f5f5f5;
  height: 1.5em;
  width: 1.5em;
  border: .1em solid rgba(0,0,0,.05);
  text-align: center;
  font: italic bold 1em/1.5em Georgia, Serif;
  color: #ccc;
  border-radius: 1.5em;
  transition: all .2s ease-out;
}

.circle-list li:hover h2:before{
  background-color: #ffd797;
  border-color: rgba(0,0,0,.08);
  border-width: .2em;
  color: #444;
  transform: scale(1.5);
}

```

### 3.4. En guise de bonus

Des animations de la numérotation ont été ajoutées au survol dans la démo ([Lien 37](#)). Malheureusement, il semble qu'actuellement, seul Firefox permet d'animer des pseudoéléments. Espérons que cela se généralisera bientôt : [Lien 40](#).

### 4. Support des navigateurs



Voir la démonstration en ligne : [Lien 37](#).

## 5. Conclusion

Merci d'avoir lu cet article. J'espère que vous l'avez apprécié. N'hésitez pas à faire part de vos commentaires.

Cet article a été publié avec l'aimable autorisation de

Catalin Rosu. L'article original peut être lu sur le site de Red Team Design ([Lien 41](#)) : CSS3 ordered list styles ([Lien 42](#)).

Retrouvez l'article de Catalin Rosu traduit par Didier Mouronval en ligne : [Lien 43](#)

## Spécifications des nouveaux sélecteurs CSS4

Oui, la spécification CSS4 est déjà en cours d'élaboration (pour les sélecteurs en tout cas). Attention, je vous rappelle qu'il s'agit d'un travail en cours, c'est-à-dire que certaines informations seront peut-être modifiées ou pourront disparaître alors que d'autres pourront s'ajouter. Mais cela vous donnera quand même une idée de ce que sera la véritable spécification CSS4. La plupart de ces idées correspondent à ce que souhaite obtenir le W3C concernant les sélecteurs CSS4.

### 1. not() et matches()



Ce sont certainement deux de mes nouveautés préférées. Vous avez peut-être déjà entendu parler du sélecteur :not(), qui a été introduit en CSS3. En revanche, le sélecteur :matches() devrait être nouveau pour vous.

#### 1.1. :not()

**E:not(s1, s2)** Un élément E qui ne correspond ni au sélecteur composé s1, ni à s2.

La spécification CSS4 précise que :not() peut être utilisé aussi bien avec des sélecteurs que des pseudoclasses. Malgré tout, il ne fonctionnera pas avec :after ou :before, mais tous les autres seront acceptés. Si vous souhaitez par exemple sélectionner tous les éléments sauf les liens, vous pouvez faire ceci :

```
*|*:not(:link):not(:visited){
  /* Règles CSS */
}
```

Utile non ? Sur des projets importants, cela va certainement permettre de réduire le code. Ce qui sera aussi le cas de :matches().

\*|\* est une règle générale de type « s'applique à tout ». Ce sélecteur permet d'appliquer le CSS à tous les éléments de la page.

#### 1.2. :matches()

**E:matches(s1, s2)** Un élément E qui correspond au sélecteur composé s1 et / ou s2.

:matches() peut s'appliquer à n'importe quelle pseudoclasse. **Vous ne pouvez pas combiner** :matches() et :not(). La spécification précise clairement que vous ne pouvez pas faire :matches(:not( ... )). De toute façon, ça n'aurait pas beaucoup de sens ! Voici un exemple qui permet de définir des règles CSS à tous les éléments

survolés par la souris :

```
*|*:matches(:hover){
  /* Règles CSS */
}
```

### 2. Nouveaux sélecteurs directs



Certains nouveaux sélecteurs sont ajoutés dans la spécification, ils ont tous leur utilité.

#### 2.1. Sélection par référence

**E /foo/ F** Un élément F relié à un élément E dont l'id est foo.

Celui-ci est un peu compliqué et il m'a fallu du temps pour bien le comprendre. Imaginons par exemple que vous ayez ceci dans un formulaire :

```
<label for="nom">Votre nom</label>
<input id="nom" type="text" />
```

L'attribut for, par définition, doit correspondre à la valeur de l'id du champ correspondant. Les deux éléments sont donc connectés. Cette pseudoclasse vous permet de cibler des éléments connectés selon ce modèle. Par exemple, si vous souhaitez cibler la balise input lorsque la souris survole la balise label, vous pouvez faire ceci :

```
label:hover /foo/ input{
  /* Règles CSS */
}
```

J'imagine que cela pourra s'appliquer à d'autres éléments connectés.

#### 2.2. Cibler les éléments parents (en CSS !)

**E! > F** Un élément E parent d'un élément F.

Celui-ci fait partie de mes favoris ! Jusqu'à présent, en CSS, le dernier élément écrit est celui que vous ciblez, par exemple dans :

```
div a{  
}
```

Si vous avez déjà utilisé les CSS, vous devez savoir que nous ciblons les balises <a> présentes dans des balises <div>. **Avec ce sélecteur, vous pouvez utiliser la même syntaxe, mais en ciblant la balise <div>.** Cela peut toutefois vous sembler inutile. Néanmoins, si vous vouliez faire cela, il suffirait d'enlever le a et cela ciblerait la balise <div>. En fait, cela permet de faire ce dont les développeurs rêvent depuis longtemps en CSS : sélectionner des éléments en fonction d'un descendant. Vous auriez pu tenter une syntaxe telle que :

```
ul li:hover < ul{  
  /* Récupérer l'élément parent ; cela ne  
  fonctionne pas */  
}
```

Avec CSS4, vous pourrez faire ceci :

```
ul! li:hover{  
  /* C'est la balise ul qui est sélectionnée */  
}
```

C'est le parent qui est sélectionné lorsqu'un élément de liste est survolé. Il s'agit d'une avancée majeure par rapport à ce que permet de faire CSS, et les possibilités offertes sont immenses.

Dans certains documents, c'est le signe \$ qui est utilisé à la place du point d'exclamation : \$E > F.

### 2.3. Casse des caractères

E[foo="bar" i]	Un élément E dont l'attribut foo correspond à bar, quelle que soit la casse de caractères.
----------------	--

C'est moins intéressant que de sélectionner les classes parentes, mais si vous avez un code tel que :

```
<pre>  
  <div class="foo"></div>  
  <div class="Foo"></div>  
  <div class="fOO"></div>  
</pre>
```

(quelle qu'en soit la raison) et que vous souhaitiez cibler toutes les <div>, vous pourriez utiliser ce sélecteur pour ça. Tout ce que vous auriez à faire serait :

```
div[class='foo' i]{  
  /* Règles CSS */  
}
```

Le petit i signifie que la casse ne doit pas être prise en compte !

### 3. Encore plus de pseudoclasses



Une bonne partie de celles-ci se rapportent aux colonnes et au nouveau sélecteur match().

#### 3.1. Les pseudoclasses de type match

E:nth-match(n)	Les éléments E n <sup>e</sup> éléments du type.
E:nth-last-match(n)	Les éléments E n <sup>e</sup> éléments du type en comptant à partir du dernier.

Comme vous l'avez peut-être remarqué, le sélecteur :match() n'est pas, en soi, particulièrement intéressant et utile. CSS4 a donc ajouté quelques sélecteurs de type nth-type. Vous pouvez ainsi sélectionner un certain nombre d'occurrences. Vous pouvez par exemple avoir envie de récupérer les quatrièmes éléments de rang d'une sélection. Pour cela, vous pouvez utiliser div:nth-match(4n). Si au contraire seul le quatrième élément vous intéresse, alors la syntaxe sera nth-match(4). L'autre possibilité est nth-last-match() qui, au lieu de compter à partir du début de la page, commencera le compte depuis la fin.

#### 3.2. Colonnes

E:column(selecteur)	Un élément E qui représente une cellule de grille ou de table appartenant à une colonne correspondant à selecteur.
E:nth-column(n)	Un élément E représentant une cellule appartenant à la colonne n d'une grille ou d'une table.
E:nth-last-column(n)	Un élément E représentant une cellule appartenant à la colonne n d'une grille ou d'une table en partant de la dernière.

Les colonnes ne servent plus uniquement pour les tables ! En essayant de finaliser la spécification pour CSS3, le W3C a introduit un certain nombre de sélecteurs de colonnes, en particulier pour les grilles. Nous avons déjà évoqué ce sujet dans ce billet : [Lien 44](#). Le sélecteur :column(selecteur) permet de sélectionner une colonne à partir de certaines caractéristiques, alors que le sélecteur :nth-column(n) permet d'en sélectionner en fonction de leur rang, et que :nth-last-column(n) en sélectionne en fonction de leur rang mais en partant de la fin.

### 4. Langages et liens



E:lang(fr, en)	Un élément E dont la langue est "fr" ou "en" (différents éléments permettent de déterminer la langue utilisée).
E:local-link	Un élément E source d'un hyperlien dont la cible est le document courant.
E:local-link(0)	Un élément E source d'un hyperlien dont la cible est le domaine courant.

L'élément de langue est susceptible (il me semble) d'être utilisé depuis CSS2, il ne s'agit donc pas d'une réelle nouveauté. On utilise ici un élément sémantique au travers du document pour appliquer des styles à des éléments désignés comme relevant d'une langue spécifique. Cela peut être utile si vous avez un site multilingue ou international.

Les sélecteurs de type :local-link seront quant à eux très utiles, en particulier en ce qui concerne la navigation. Jusqu'à présent, nous devions utiliser des classes spécifiques à appliquer pour les liens internes, qu'il faut ajouter soit en PHP soit manuellement. **Avec :local-link, nous pouvons cibler directement les liens internes à la page en cours.** Si vous souhaitez par exemple appliquer des styles à un menu contenant des ancres, il vous suffit de faire :

```
#nav:local-link{
  box-shadow : 0px 0px 10px rgba(0,0,0,0.4) ;
}
```

Plutôt simple, non ? Mais on peut aller un peu plus loin en précisant l'arborescence de l'URL. Par exemple :

```
#nav:local-link(0){
}
```

va cibler tous les liens vers la racine du site (dans notre cas, <http://inserthtml.developpez.com/>).

Si votre URL est par exemple <http://www.inserthtml.com/2011/03/25/name/>, alors

- :local-link(0) va cibler les URL vers <http://www.inserthtml.com/> ;
- :local-link(1) va cibler les URL vers <http://www.inserthtml.com/2011/> ;
- :local-link(2) va cibler les URL vers <http://www.inserthtml.com/2011/03/> ;
- :local-link(3) va cibler les URL vers <http://www.inserthtml.com/2011/03/25/> ;
- :local-link(4) va cibler les URL vers <http://www.inserthtml.com/2011/03/25/name/>.

## 5. Maîtriser le temps !



E:current	Un élément E présent dans un contexte temporel.
E:current(s)	Un élément E qui est le plus proche

	élément :current correspondant au sélecteur s.
E:past	Un élément E passé dans un contexte temporel.
E:future	Un élément E futur dans un contexte temporel.

Aussi prometteur que cela puisse sembler, CSS ne vous permet pas actuellement de manipuler le continuum espace/temps de quelque façon que ce soit (faudra-t-il attendre CSS5 ?). Ce que ces éléments permettent est de gérer les styles en temps réel. Par exemple, mettre en valeur un paragraphe en cours de lecture (en utilisant la spécification vocale de CSS3), ou de donner plus de possibilités de gestion de vidéo en HTML5 (comme les sous-titres) et peut-être plus lorsque les spécifications CSS4 seront finalisées.

## 6. Interface utilisateur (comprenant CSS3UI)



E:indeterminate	Un élément E dans un état indéterminé (ni sélectionné ni non sélectionné).
E:default	Un élément E ayant sa valeur par défaut.
E:in-range et E:out-of-range	Un élément E dans ou hors d'un intervalle.
E:required et E:optional	Un élément E requis ou optionnel.
E:read-only et E:read-write	Un élément E pouvant, ou non, être édité.

Il s'agit à chaque fois d'éléments d'interface utilisateur. Il n'y a pas besoin de beaucoup plus d'explications. Ils ne s'appliquent essentiellement, pour l'heure, qu'aux éléments de formulaires, en se basant sur leur état. Mais à l'avenir de nouveaux éléments seront peut-être aussi concernés, ce qui rendra ces sélecteurs encore plus utiles. Ils sont apparus avec CSS3, mais on peut imaginer que CSS4 en ajoutera encore d'autres.

## 7. Conclusion

La spécification CSS4, dans son état actuel, apporte déjà beaucoup de ce dont les développeurs Web ont rêvé depuis longtemps. Nous ne pouvons qu'espérer que les navigateurs ne tarderont pas à implémenter ces nouveautés au plus vite (en particulier, le ciblage des éléments parents). Personnellement, je suis impatient que cela arrive. Et vous, qu'en pensez-vous ?

Cet article a été traduit avec l'aimable autorisation de inserthtml ([Lien 45](#)). L'article original : The Brand New CSS4 Selectors Specification peut être vu sur le site de inserthtml : [Lien 46](#).

Retrouvez l'article de Johnny Simpson traduit par Didier Mouronval en ligne : [Lien 47](#)

## Créez des formulaires fabuleux avec HTML5

Jusqu'à présent, les formulaires ne provoquaient pas un enthousiasme excessif, mais HTML5 apporte des améliorations importantes, autant pour les développeurs qui les créent que pour les utilisateurs qui les remplissent. Ces nouveautés concernent les éléments de formulaires, les attributs, les types de champs, la validation par le navigateur, des styles CSS3 et l'objet FormData qui rendent l'utilisation des formulaires plus agréable et conviviale.

### 1. Compatibilité des navigateurs

Au moment de la rédaction de cet article (NDT Juin 2011), le support des nouveaux éléments et champs de formulaire ainsi que leurs attributs varie largement en fonction des navigateurs. Même parmi ceux supportant une fonctionnalité donnée, le comportement peut être différent d'un navigateur à l'autre. En fait, le support des formulaires HTML5 est en constante et rapide évolution. Lors de la rédaction de cet article, ces tableaux étaient les plus à jour que j'ai pu trouver : [Lien 48](#).

### 2. Récapitulatif des nouveautés

#### 2.1. Nouveaux éléments

HTML5 introduit cinq nouveaux éléments de formulaire.

Élément	Rôle	Notes
progress	Représente la progression d'une opération.	Par exemple, l'élément progress peut correspondre à la progression de l'envoi d'un fichier.
meter	Représente une mesure scalaire dans un intervalle déterminé.	L'élément meter peut servir par exemple à représenter une valeur comme une température ou un poids.
datalist	Représente un ensemble d'options prédéfinies qui peuvent être utilisées avec le nouvel attribut list de la balise input pour créer un menu déroulant de valeurs possibles.	Lorsque l'input lié à datalist prend le focus, les suggestions correspondant aux options apparaissent.
keygen	Permet la génération de clés.	Lorsque le formulaire est soumis, la clé privée est stockée par le navigateur et la clé publique est envoyée au serveur.
output	Affiche le résultat d'un calcul.	Peut par exemple servir à afficher la somme de deux champs numériques.

#### 2.2. Nouveaux types de champs

HTML5 introduit treize nouveaux types de champs. S'ils

ne sont pas reconnus par le navigateur, celui-ci les affichera comme des champs texte simples.

Type	Rôle	Notes
tel	Pour renseigner un numéro de téléphone.	L'attribut tel ne définit pas de format spécifique, si vous souhaitez forcer un format particulier, vous devrez utiliser l'attribut pattern ou la méthode setCustomValidity().
search	Permet d'indiquer à l'utilisateur qu'il s'agit d'un champ de recherche.	La différence avec un input de type text est purement stylistique. L'utilisation d'un type search va permettre au navigateur d'afficher le champ dans un style approprié.
url	Pour renseigner une URL.	Ce type de champ est destiné à une unique URL absolue ( <a href="#">Lien 49</a> ), ce qui laisse un large choix d'entrées possibles.
email	Pour renseigner une ou plusieurs adresses e-mail.	Si l'attribut multiple est présent, plusieurs adresses, séparées par des virgules, peuvent être entrées.
datetime	Pour renseigner une date et une heure sur le fuseau UTC.	
date	Pour renseigner une date.	
month	Pour renseigner une date avec une année et un mois.	
week	Pour renseigner une date à partir du numéro de semaine de l'année considérée.	Un exemple serait 2011-W05 pour la cinquième semaine de 2011.
time	Pour renseigner une heure comprenant l'heure, les minutes, les secondes et les	

	fractions de seconde.	
datetime-local	Pour renseigner une date et une heure sans considération de fuseau.	
number	Pour renseigner une valeur numérique.	Les valeurs sont des nombres à virgule flottante ( <a href="#">Lien 50</a> ).
range	Pour renseigner une valeur numérique dans un intervalle donné.	L'implémentation pour les navigateurs supportant ce type est un curseur.
color	Pour renseigner une couleur.	La valeur doit être une valeur de couleur valide en minuscules ( <a href="#">Lien 51</a> ).

### 2.3. Nouveaux attributs

De nouveaux attributs concernant les formulaires et leurs éléments ont aussi été introduits par HTML5.

Attribut	Rôle	Notes
autofocus	Donne le focus à l'élément au chargement de la page.	autofocus peut être appliqué aux balises input, select, textarea et button.
placeholder	Donne une indication à l'utilisateur sur le type de donnée à entrer.	La valeur de l'attribut placeholder est affichée jusqu'à ce que le champ prenne le focus ainsi qu'à la perte du focus si le champ est vide. S'applique aux balises input et textarea.
form	Précise à quel(s) formulaire(s) est attaché le champ.	En utilisant l'attribut form, le champ peut être placé n'importe où dans la page, même en dehors des balises <form>. Un même champ peut ainsi être associé à plusieurs formulaires.
required	Attribut booléen indiquant que le champ est obligatoire.	Cet attribut sert essentiellement pour les vérifications côté navigateur sans utiliser JavaScript.
autocomplete	Attribut booléen indiquant si les suggestions de saisies précédentes doivent être affichées.	Le navigateur retient les saisies associées à un champ et les propose en suggestion lors de la

		saisie, cet attribut permet de désactiver cette fonctionnalité.
pattern	Permet de préciser une expression régulière que doit satisfaire la saisie.	Lorsque vous utilisez cet attribut, n'oubliez pas d'ajouter un attribut title pour indiquer à l'utilisateur le format attendu.
dirname	La valeur de cet attribut permet d'envoyer à la soumission du formulaire un paramètre indiquant le sens d'écriture.	Le sens d'écriture peut être "ltr" ou "rtl". Cette valeur est envoyée avec les autres champs du formulaire si l'attribut est précisé.
novalidate	Désactive les vérifications du formulaire avant de l'envoyer.	
formaction	Permet de modifier l'attribut action du formulaire.	Cet attribut est valable pour les éléments input (de type submit) et button.
formenctype	Permet de modifier l'attribut enctype du formulaire.	Cet attribut est valable pour les éléments input (de type submit) et button.
formmethod	Permet de modifier l'attribut method du formulaire.	Cet attribut est valable pour les éléments input (de type submit) et button.
formnovalidate	Permet de modifier l'attribut novalidate du formulaire.	Cet attribut est valable pour les éléments input (de type submit) et button.
formtarget	Permet de modifier l'attribut target du formulaire.	Cet attribut est valable pour les éléments input (de type submit) et button.

### 2.4. L'objet FormData

L'une des améliorations apportées à XMLHttpRequest ([Lien 52](#)) est l'ajout de l'objet FormData ([Lien 53](#)). Avec cet objet, vous pouvez créer et envoyer des paires clé/valeur (optionnellement, des fichiers) avec XMLHttpRequest. Avec cette technique, vous pouvez envoyer des données au même format que si vous aviez validé un formulaire à l'aide de la méthode submit() de l'objet form avec l'encodage multipart/form-data.

L'objet FormData vous permet donc de créer des formulaires à la volée avec JavaScript puis de les

soumettre en utilisant XMLHttpRequest.send(). Voici un exemple d'utilisation :

```
var formData = new FormData();
formData.append("part_num", "123ABC");
formData.append("part_price", 7.95);
formData.append("part_image", somefile)

var xhr = new XMLHttpRequest();
xhr.open("POST", "http://some.url/");
xhr.send(formData);
```

Vous pouvez aussi utiliser FormData pour ajouter des données à un formulaire existant avant de le soumettre :

```
var formElement =
document.getElementById("someFormElement");
var formData = new FormData(formElement);
formData.append("part_description", "The best
part ever!");

var xhr = new XMLHttpRequest();
xhr.open("POST", "http://some.url/");
xhr.send(formData);
```

### 3. Validation par le navigateur

Soyons honnêtes., la validation des données d'un formulaire est une tâche assez pénible, mais nous devons malgré tout la faire. Actuellement, pour valider les données d'un formulaire côté client, vous avez probablement codé vos propres fonctions JavaScript ou utilisé une bibliothèque. Cela afin de vérifier la validité du format de certaines entrées ou que les champs obligatoires ont bien été remplis avant d'envoyer les données au serveur.

Les nouveaux attributs comme required ou pattern, utilisés conjointement avec les pseudoclasses CSS facilitent grandement ces vérifications et l'affichage des informations utiles à l'utilisateur. D'autres techniques plus avancées permettent d'utiliser JavaScript pour définir des règles personnalisées de validation et les messages associés, ou de déterminer si un élément est invalide et pourquoi.

#### 3.1. L'attribut required

Si l'attribut required est présent, alors le champ en question doit contenir une valeur lorsque le formulaire est soumis. Voici un exemple de champ input concernant une adresse e-mail obligatoire et permettant de vérifier que l'adresse est bien renseignée et valide selon les critères définis ici ([Lien 54](#)) :

```
<input type="email" id="email_addr"
name="email_addr" required />
```

#### 3.2. L'attribut pattern

L'attribut pattern permet de définir une expression régulière qui sera utilisée pour déterminer si la saisie est conforme au format attendu. L'exemple suivant correspond à un champ texte requis et représentant un code particulier. Dans l'exemple, nous souhaitons que ce code soit composé de trois lettres majuscules suivies de quatre chiffres. Les attributs required et pattern permettent de s'assurer que le

champ sera rempli au format souhaité lors de la soumission. Enfin, lors du survol du champ, l'utilisateur verra apparaître un message correspondant à l'attribut title lui indiquant le format attendu.

```
<input type="text" id="part" name="part" required
pattern="[A-Z]{3}[0-9]{4}"
title="Part numbers consist of 3 uppercase
letters followed by 4 digits."/>
```

À partir de l'exemple précédent, on peut ajouter une bordure rouge à l'input tant que le champ ne contient pas de saisie valide. Pour cela, nous ajoutons le code CSS suivant pour créer la bordure rouge si le champ est invalide :

```
:invalid {
border: 2px solid #ff0000;
}
```

#### 3.3. L'attribut formnovalidate

L'attribut formnovalidate s'applique aux input et button de type submit. S'il est présent, alors la validation par le navigateur sera désactivée. Dans l'exemple suivant, la validation du formulaire est effectuée en cliquant sur "Submit" mais ne l'est pas en cliquant sur "Save".

```
<input type="text" id="part" name="part" required
pattern="[A-Z]{3}[0-9]{4}"
title="Part numbers consist of 3 uppercase
letters followed by 4 digits."/>
<input type="submit" formnovalidate value="Save">
<input type="submit" value="Submit">
```

#### 3.4. L'API de validation

L'API de validation ([Lien 55](#)) met à votre disposition des outils puissants permettant de gérer une validation personnalisée des formulaires. Cette API permet de déterminer des messages d'erreurs, de vérifier si un élément est valide et de préciser la raison pour laquelle il est invalide. Dans l'exemple suivant, nous affichons un message personnalisé si les valeurs de deux champs sont différentes.

```
<label>Email:</label>
<input type="email" id="email_addr"
name="email_addr">

<label>Repeat Email Address:</label>
<input type="email" id="email_addr_repeat"
name="email_addr_repeat" oninput="check(this)">

<script>
function check(input) {
if (input.value !=
document.getElementById('email_addr').value) {
input.setCustomValidity('The two email
addresses must match.');
```

## 4. Un exemple récapitulatif

Voici un exemple ([Lien 56](#)) de formulaire de réservation qui permet d'assembler les différents éléments déjà vus : nouveaux types d'input et leurs attributs, validation de formulaire et mise en forme CSS avec les sélecteurs adaptés.

Nom complet :

Adresse e-mail :

Confirmez l'adresse e-mail :

Date d'arrivée :

Nombre de nuitées (99 € par nuit) :

Nombre d'invités (10 € par invité supplémentaire) :

Total : 99.00 €

Code de promotion :

[Voir le formulaire en ligne](#)

Voici les codes HTML, CSS et JavaScript du formulaire :

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8" />
  <title>Formulaire HTML5</title>
  <style>
    body{
      background-image: url(background.png);
      font-family: 'Open Sans', sans-serif;
      font-weight: 400;
      font-size: 13px;
    }

    :invalid {
      background-color: #F0DDDD;
      border-color: #e88;
      -webkit-box-shadow: 0 0 5px rgba(255,
0, 0, .8);
      -moz-box-shadow: 0 0 5px rbb(255, 0,
0, .8);
      -o-box-shadow: 0 0 5px rbb(255, 0,
0, .8);
      -ms-box-shadow: 0 0 5px rbb(255, 0, 0,
.8);
      box-shadow:0 0 5px rgba(255, 0, 0, .8);
    }

    :required {
      border-color: #88a;
      -webkit-box-shadow: 0 0 5px rgba(0, 0,
255, .5);
      -moz-box-shadow: 0 0 5px rgba(0, 0,
255, .5);
      -o-box-shadow: 0 0 5px rgba(0, 0,
255, .5);
    }
  </style>
</head>
<body>
  <form oninput="total.value =
(nights.valueAsNumber * 99) +
((guests.valueAsNumber - 1) * 10)">
    <label>Nom complet :</label>
    <input type="text" id="full_name"
name="full_name" placeholder="Jane Doe" required>
    <label>Adresse e-mail :</label>
    <input type="email" id="email_addr"
name="email_addr" required>
    <label>Confirmez l'adresse e-mail :</label>
    <input type="email" id="email_addr_repeat"
name="email_addr_repeat" required
oninput="check(this)">
    <label>Date d'arrivée :</label>
    <input type="date" id="arrival_dt"
name="arrival_dt" required>
    <label>Nombre de nuitées (99 € par nuit)
:</label>
    <input type="number" id="nights"
name="nights" value="1" min="1" max="30"
required>
    <label>Nombre d'invités (10 € par invité
supplémentaire) :</label>
```

```

    <input type="text" value="Jane Doe" />
    <input type="text" />
    <input type="text" />
    <input type="text" />
    <input type="text" value="1" />
    <input type="text" value="1" />
    <input type="text" />
    <input type="submit" value="Effectuer la réservation" />
  </form>
</body>
</html>
```

```

    .5);
    box-shadow: 0 0 5px rgba(0, 0, 255, .
5);
  }

  form {
    width:300px;
    margin: 20px auto;
  }

  input {
    font-family: "Helvetica Neue",
Helvetica, Arial, sans-serif;
    border:1px solid #ccc;
    font-size:20px;
    width:300px;
    min-height:30px;
    display:block;
    margin-bottom:15px;
    margin-top:5px;
    outline: none;

    -webkit-border-radius:5px;
    -moz-border-radius:5px;
    -o-border-radius:5px;
    -ms-border-radius:5px;
    border-radius:5px;
  }

  input[type=submit] {
    background:none;
    padding:10px;
    cursor: pointer;
  }

  label{
    cursor: pointer;
  }
</style>
</head>
<body>
  <form oninput="total.value =
(nights.valueAsNumber * 99) +
((guests.valueAsNumber - 1) * 10)">
    <label>Nom complet :</label>
    <input type="text" id="full_name"
name="full_name" placeholder="Jane Doe" required>
    <label>Adresse e-mail :</label>
    <input type="email" id="email_addr"
name="email_addr" required>
    <label>Confirmez l'adresse e-mail :</label>
    <input type="email" id="email_addr_repeat"
name="email_addr_repeat" required
oninput="check(this)">
    <label>Date d'arrivée :</label>
    <input type="date" id="arrival_dt"
name="arrival_dt" required>
    <label>Nombre de nuitées (99 € par nuit)
:</label>
    <input type="number" id="nights"
name="nights" value="1" min="1" max="30"
required>
    <label>Nombre d'invités (10 € par invité
supplémentaire) :</label>
```

```

<input type="number" id="guests"
name="guests" value="1" min="1" max="4" required>

<label>Total :</label>
<output id="total"
name="total">99</output>.00 €
<br><br>

<label>Code de promotion :</label>
<input type="text" id="promo" name="promo"
pattern="[A-Za-z0-9]{6}"
title="Le code de promotion contient six
caractères alphanumériques.">

<input type="submit" value="Effectuer la
réservation" />
</form>

<script>

```

```

function check(input) {
    if (input.value !=
document.getElementById('email_addr').value) {
        input.setCustomValidity('Les deux
adresses e-mail ne correspondent pas.');
```

```

    } else {
        // le champ est valide : on réinitialise
le message d'erreur
        input.setCustomValidity('');
    }
}
</script>
</body>
</html>

```

Voir l'exemple en ligne : [Lien 56](#).

Retrouvez l'article de Jan Kleinert traduit par Didier Mouronval en ligne : [Lien 57](#)



### Oracle entame la migration vers Java 7 pour le grand public et publie Java SE 7 Update 4 et JavaFX 2.1

Java 7 est désormais disponible pour le grand public.

Publiée pour les développeurs depuis juillet 2011, la dernière version du langage Java est désormais disponible dans le Java Runtime Environment (JRE) pour les consommateurs.

Oracle a annoncé récemment qu'il a commencé le processus de mise à jour des utilisateurs vers la version 7. Les personnes utilisant le runtime Java vont donc être migrées automatiquement vers la version 7 progressivement au cours des prochains mois.

La société encourage les utilisateurs des anciennes versions à utiliser JRE 7 qui contient les fonctionnalités les plus récentes, des améliorations de performance et des mises à jour de sécurité.

Java 7 est disponible pour les utilisateurs sur Java.com, ils peuvent vérifier la version de Java qu'ils utilisent sur ce site. Oracle recommande de désinstaller l'ancienne version avant de migrer sur la récente.

Cette décision intervient juste quelques jours après la publication de Java Standard Edition 7 Update 4 (Java SE 7 Update 4) et de JavaFX 2.1 ([Lien 58](#)).

Java SE 7 Update 4 tire parti des meilleures caractéristiques des machines virtuelles Oracle JRockit et Oracle Java HotSpot. Toutes les améliorations de performances disponibles dans Oracle JRockit ont été fusionnées dans Oracle Java HotSpot et OpenJDK, l'implémentation open source de Java SE.

Cette mise à jour intègre également la nouvelle génération de l'algorithme de récupération d'espace mémoire Garbage First (G1).

La plateforme de création d'applications internet riches (RIA) JavaFX 2.1 introduit de nouvelles fonctionnalités comme le support des médias numériques au format MPEG-4 utilisant de la vidéo H.264/AVC et du son AAC (Advanced Audio Coding), la prise en charge des appels aux méthodes Java depuis JavaScript dans WebView permettant le rendu du contenu HTML/JavaScript en laissant JavaScript appeler des API Java afin de se décharger des opérations spécifiques au langage...

Oracle a par ailleurs publié la première version du JDK pour Java 7 pour Mac OS X.

**Télécharger Java SE 7 Update 4 :** [Lien 59](#)

**Télécharger JavaFX 2.1 :** [Lien 60](#)

*Commentez la news de Hinault Romaric en ligne : [Lien 61](#).*

## Les derniers tutoriels et articles

### JGraphX : Les bases

JGraphX est une API qui permet de dessiner des graphes dans une application Java. Le principal reproche qui est fait contre cette API est le manque de documentation. Ce tutoriel va vous apporter les bases pour prendre en main cette API.

#### 1. Introduction

##### 1.1. JGraphX : c'est quoi

JGraphx est une bibliothèque Java qui permet de dessiner des graphes en 2D dans une application. Vous allez dire, c'est quoi un graphe. Pour cela, il suffit de lire le billet sur Wikipédia qui décrit la théorie des graphes à cette adresse : [Lien 62](#).

Ce n'est pas pour vous faire peur, mais maintenant vous allez comprendre la raison du développement de cette bibliothèque.

Dans ce tutoriel, JGraphX n'est pas là pour résoudre des parcours de graphes, mais va plutôt servir à les dessiner, les représenter schématiquement et fournir un modèle de

données qui va permettre de parcourir le graphe qui a été dessiné.

##### 1.2. Deux versions mxGraph et JGraphx

À l'origine du projet, il existait JGraph, une bibliothèque Java gratuite permettant de dessiner et parcourir des graphes. Deux versions distinctes ont été créées.

- JGraphx est la version gratuite du produit. Elle permet de dessiner des graphes dans une application Java. C'est cette version qui sera décrite dans le tutoriel.
- mxGraph, quant à elle, permet de dessiner des graphes dans des applications Web, tel que c'est décrit dans le site Web de JGraph. Vous avez ici ([Lien 63](#)), un exemple d'application. Cette version n'est pas gratuite, il faut bien évidemment que son

auteur puisse vivre.

### 1.3. Téléchargement de JGraphX

Le téléchargement de JGraphX se fait ici : [Lien 64](#).

Le fichier à télécharger est un fichier ZIP que vous pouvez décompresser sur votre disque et est constitué ainsi :

- le répertoire doc contient la documentation de JGraphX et est constitué d'un manuel utilisateur (assez rudimentaire) et de la Javadoc' ;
- le répertoire exemples contient quelques applications pour comprendre comment utiliser cette bibliothèque, et je peux vous assurer que c'est bien utile' ;
- le répertoire lib contient la bibliothèque Java que vous devez inclure dans le classpath pour pouvoir faire une application basée sur JGraphX' ;
- le répertoire src contient les sources de la bibliothèque. Cela peut aider aussi pour mieux comprendre comment elle fonctionne.

## 2. Projet Java pour JGraphX

Afin de pouvoir utiliser la bibliothèque JGraphX, il suffit simplement de mettre le fichier jgraphx.jar dans le classpath de l'application. Ce jar se trouve dans le répertoire lib du ZIP que l'on a téléchargé.

### 3. Une première application

#### 3.1. Comprendre le code de l'application

Pour débiter, on va créer une petite application simple qui affiche un "hello World" dans une JFrame. Je ne l'ai pas écrite, elle provient directement du manuel.

#### UN premier exemple

```
package com.bpy.jgraph.tutorial;

import javax.swing.JFrame;

import com.mxgraph.model.mxCell;
import com.mxgraph.model.mxGeometry;
import com.mxgraph.swing.mxGraphComponent;
import com.mxgraph.view.mxGraph;

public class JGraphExemple1 extends JFrame {

    /** Pour éviter un warning venant du JFrame */
    private static final long serialVersionUID =
    -8123406571694511514L;

    public JGraphExemple1() {
        super("JGraphX tutorial: Exemple 1");

        mxGraph graph = new mxGraph();
        Object parent = graph.getDefaultParent();

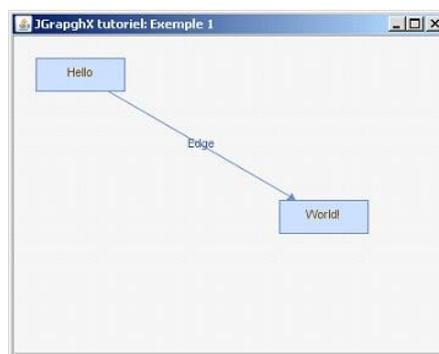
        graph.getModel().beginUpdate();
        try {
            Object v1 = graph.insertVertex(parent,
            null, "Hello", 20, 20, 80, 30);
            Object v2 = graph.insertVertex(parent,
            null, "World!", 240, 150, 80, 30);
            graph.insertEdge(parent, null, "Edge", v1,
```

```
v2);
        } finally {
            graph.getModel().endUpdate();
        }

        mxGraphComponent graphComponent = new
        mxGraphComponent(graph);
        getContentPane().add(graphComponent);
    }

    /**
     * @param args
     */
    public static void main(String[] args) {
        JGraphExemple1 frame = new JGraphExemple1();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON
        _CLOSE);
        frame.setSize(400, 320);
        frame.setVisible(true);
    }
}
```

L'exécution de ce code permet d'obtenir la fenêtre suivante.



Que voit-on dans cette fenêtre ?

- Deux rectangles 'Hello' et 'World' de couleur bleue ; ces deux éléments sont nommés vertex dans le langage JGraphX.
- Une flèche qui relie ces deux éléments. Cette flèche est un Edge et on peut remarquer aussi que cette flèche est aussi nommée. La flèche indique, dans ce cas, la destination.

Eh bien, ce n'est pas tout, car il existe aussi un élément non visible qui est le graphe par lui-même. C'est le parent des trois éléments que l'on voit.

En regardant de plus près le code.

```
mxGraph graph = new mxGraph();
```

Tout graphe doit commencer par ceci. C'est le constructeur du graphe.

	<b>other</b>	<b>0</b>	<b>1</b>	
Object parent = graph.getDefaultParent();				

On commence par récupérer le point d'entrée du graphe. Pour en comprendre la raison, il faut imaginer que votre graphe se présente sous la forme d'une structure de données de type arbre. Dans un arbre, il existe toujours une racine et ce parent est la racine de votre graphe.

other	0	1	
Object v1 = graph.insertVertex(parent, null, "Hello", 20, 20, 80, 30);			
Object v2 = graph.insertVertex(parent, null, "World!", 240, 150, 80, 30);			

Ici, on crée les deux vertex (rectangles) en définissant leurs propriétés. Explication des paramètres.

- parent : les vertex sont liés directement à la racine du graphe.
- null : ici on attend une chaîne de caractères qui sert d'identifiant pour le vertex, cette chaîne est optionnelle, on peut ne pas la renseigner.
- Objet contenu par le vertex. Ici c'est une string mais, cela pourrait être n'importe quel objet, tout dépend de ce que l'on veut faire avec ce graphe après.
- Les quatre derniers paramètres correspondent aux x, y, largeur et hauteur du rectangle.

other	0	1	
graph.insertEdge(parent, null, "Edge", v1, v2);			

Et ici, on ajoute le lien entre les deux vertex, ce que l'on nomme un edge.

- parent : les vertex sont liés directement à la racine du graphe
- null : ici on attend une chaîne de caractères qui sert d'identifiant pour ce edge, cette chaîne est optionnelle, on peut ne pas la renseigner.
- 'Edge' : un objet contenu par le vertex, ici un String.
- V1 est le vertex source du edge.
- V2 est vertex destination du edge.

Pour finir, on voit que la création du graphe est encapsulée dans le code suivant :

other	0	1	
graph.getModel().beginUpdate();			
try {			
'€ ..			
} finally {			
graph.getModel().endUpdate();			
}			

Cette encapsulation bloque l'affichage du graphe pendant la modification du modèle. Cette protection est basée sur le principe d'un compteur, beginUpdate incrémente le compteur, endUpdate le décrémente et l'affichage n'est possible que si le compteur vaut 0.

### 3.2. Comportement des objets dessinés

Jouons un peu avec notre graphe.

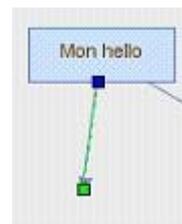
Si on place le curseur de la souris sur un vertex, on peut voir que le curseur réagit différemment en fonction de sa position sur le vertex.

- Vers le centre du vertex on a le curseur en forme de main. Cela indique que vous pouvez faire de

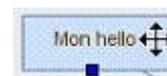
l'édition sur cet objet.



- Double-clic permet de modifier le texte dans la boîte. Pour valider le nouveau texte, il suffit de cliquer ailleurs dans le graphe.
- Simple clic permet de rajouter un edge à partir de ce vertex. Pour cela, clic gauche sans relâcher le bouton de la souris et ensuite déplacer la souris dans le graphe. Au relâchement de la souris, un nouvel edge est créé.



- Proche du bord, on a un curseur en forme de quatre flèches. Ce curseur indique que l'on peut modifier la position et la taille du vertex.

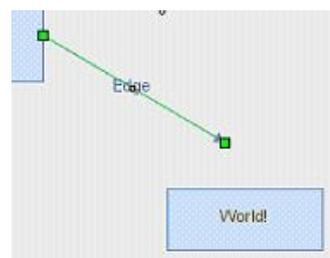


- Un simple clic permet de sélectionner le vertex, son affichage est modifié pour indiquer les points d'action possibles (petits rectangles aux coins et centre du contour). En utilisant ces points d'action, vous pouvez modifier la taille du vertex. Si on ne clique pas dans les points d'action, on peut déplacer le vertex dans le graphe. On peut remarquer aussi que l'edge qui lie les deux vertex suit le mouvement et reste connecté aux vertex.

L'edge est aussi modifiable, comme pour le vertex. On peut le sélectionner en cliquant dessus.

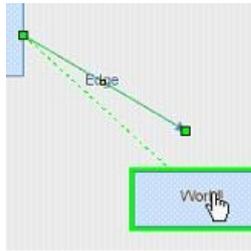


Dans ce cas, on remarque qu'un edge qui est sélectionné affiche deux petits rectangles à ses extrémités. La couleur de ces rectangles est importante, car elle indique que l'edge est bien relié aux vertex. En cliquant sur l'edge, on peut le déplacer et obtenir ceci.

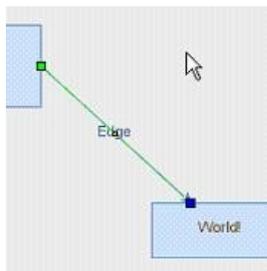


On voit que les petits rectangles ont changé de couleur, ils

sont maintenant verts. Cela signifie qu'ils ne sont plus liés aux vertex. Pour le vérifier, il suffit de bouger un vertex pour s'apercevoir que l'edge ne le suit pas. Pour reconnecter votre edge au vertex, il suffit de ramener un des petits rectangles vers le centre du vertex pour le reconnecter comme ceci :



et on obtient' :



En conclusion, on s'aperçoit que ce simple bout de code inclut déjà un grand nombre de mécanismes complexes pour le dessin des graphes.

### 3.3. Compréhension du modèle de donnée

JGraphX est basé sur le principe Modèle/Vue, c'est-à-dire que vous voyez la représentation graphique d'un modèle. Ce modèle est basé sur une structure de données de type arbre. Afin de visualiser l'arbre de données qui a servi on va ajouter le code suivant à notre application.

```
public JGraphExemple1() {
    'e|..
    displayModel((mxCell) parent, "");
}

private void displayModel(mxCell cell, String indent) {
    System.out.println(indent+cell.getValue()
    + "("+cell.getClass().getName()+")");
    int nbChilds = cell.getChildCount();
    indent = indent + " ";
    for (int i=0; i<nbChilds; i++) {
        displayModel((mxCell) cell.getChildAt(i),
        indent);
    }
}
```

La méthode displayModel va permettre d'afficher le contenu de notre modèle de graph.

Avec notre modèle, on va obtenir l'affichage suivant :

```
null (com.mxgraph.model.mxCell)
Hello (com.mxgraph.model.mxCell)
World! (com.mxgraph.model.mxCell)
Edge (com.mxgraph.model.mxCell)
```

Ceci montre une architecture avec un élément null (c'est l'élément parent) qui contient trois enfants (Hello, World! et Edge).

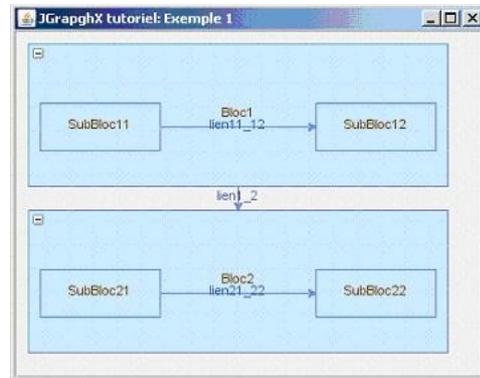
Maintenant, on va compliquer un peu notre graphe pour voir comment notre modèle va être impacté.

#### Un modèle plus complexe

```
graph.getModel().beginUpdate();
try {
    Object level1 = graph.insertVertex(parent,
    null, "Bloc1", 10, 10, 350, 120);
    Object level2 = graph.insertVertex(parent,
    null, "Bloc2", 10, 150, 350, 120);
    Object level1_1 = graph.insertVertex(level1,
    null, "SubBloc11", 10, 50, 100, 40);
    Object level1_2 = graph.insertVertex(level1,
    null, "SubBloc12", 240, 50, 100, 40);
    Object level2_1 = graph.insertVertex(level2,
    null, "SubBloc21", 10, 50, 100, 40);
    Object level2_2 = graph.insertVertex(level2,
    null, "SubBloc22", 240, 50, 100, 40);

    graph.insertEdge(level1, null, "lien11_12",
    level1_1, level1_2);
    graph.insertEdge(level2, null, "lien21_22",
    level2_1, level2_2);
    graph.insertEdge(parent, null, "lien1_2",
    level1, level2);
}
finally {
    graph.getModel().endUpdate();
}
```

On va alors obtenir le graphe suivant:



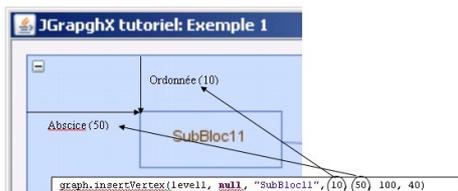
et l'affichage du modèle donnera :

```
null (com.mxgraph.model.mxCell)
Bloc1 (com.mxgraph.model.mxCell)
SubBloc11 (com.mxgraph.model.mxCell)
SubBloc12 (com.mxgraph.model.mxCell)
lien11_12 (com.mxgraph.model.mxCell)
Bloc2 (com.mxgraph.model.mxCell)
SubBloc21 (com.mxgraph.model.mxCell)
SubBloc22 (com.mxgraph.model.mxCell)
lien21_22 (com.mxgraph.model.mxCell)
lien1_2 (com.mxgraph.model.mxCell)
```

On voit que nous avons ajouté un nouveau niveau à notre modèle (Root, enfants, petits-enfants).

Du point de vue comportement graphique, on peut noter les points suivants :

- les coordonnées d'un élément du graphe sont toujours données par rapport à son parent ;



- lorsque l'on déplace un bloc parent, les blocs enfants suivent le mouvement.

#### 4. Comment agir sur le comportement du graphe

Ce chapitre va expliquer les possibilités offertes par JGraphX pour customiser les actions que l'on peut faire sur le graphe. Par exemple, on peut souhaiter ne pas pouvoir éditer le texte dans un Vertex ou un Edge, interdire les déplacements, etc.

##### 4.1. Comportement général du graphe

Si on prend, par exemple, le cas du déplacement, JGraphX offre la possibilité d'autoriser ou d'interdire le déplacement des objets d'une manière globale ou individuelle.

Pour interdire globalement le déplacement de tous les objets dans un graphe, c'est au niveau graphe que l'on doit indiquer notre choix. Pour cela, il faut utiliser la méthode `setCellsMovable` comme ceci :

```
public JGraphExemple1() {
    super("JGraphX tutorial: Exemple 1");

    mxGraph graph = new mxGraph();
    Object parent = graph.getDefaultParent();

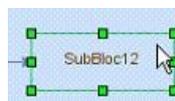
    graph.getModel().beginUpdate();
    try
    {
        Object level1 = graph.insertVertex(parent,
            null, "Bloc1", 10, 10, 350, 120);
        Object level2 = graph.insertVertex(parent,
            null, "Bloc2", 10, 150, 350, 120);
        Object level1_1 = graph.insertVertex(level1,
            null, "SubBloc1", 10, 50, 100, 40);
        Object level1_2 = graph.insertVertex(level1,
            null, "SubBloc12", 240, 50, 100, 40);
        Object level2_1 = graph.insertVertex(level2,
            null, "SubBloc21", 10, 50, 100, 40);
        Object level2_2 = graph.insertVertex(level2,
            null, "SubBloc22", 240, 50, 100, 40);

        graph.insertEdge(level1, null, "lien11_12",
            level1_1, level1_2);
        graph.insertEdge(level2, null, "lien21_22",
            level2_1, level2_2);
        graph.insertEdge(parent, null, "lien_1_2",
            level1, level2);
    }
    finally
    {
        graph.getModel().endUpdate();
    }

    graph.setCellsMovable(false);
}
```

```
mxGraphComponent graphComponent = new
mxGraphComponent(graph);
getContentPane().add(graphComponent);
}
```

Si on relance l'application, on peut vérifier que les objets dans le graphe ne peuvent plus se déplacer. On peut aussi remarquer que le curseur de la souris n'est plus modifié.



##### 4.2. Comportement individuel des objets

On sait maintenant modifier le comportement en général des objets d'un graphe. Mais on peut aussi vouloir modifier le comportement de quelques objets seulement. JGraphX nous offre aussi cette possibilité en jouant sur les styles des objets.

On souhaite, par exemple, ne pas pouvoir bouger les vertex `level1` et `level2` tout en autorisant le déplacement des vertex qui sont contenus par ces deux vertex parents.

Pour cela, lorsque l'on crée ces deux vertex, on ajoute un style comme ceci.

```
String style = mxConstants.STYLE_MOVABLE + "=0";
Object level1 = graph.insertVertex(parent, null,
"Bloc1", 10, 10, 350, 120, style);
Object level2 = graph.insertVertex(parent, null,
"Bloc2", 10, 150, 350, 120, style);
```

Quand on relance l'application, on peut vérifier que ces deux objets ne peuvent plus se déplacer, par contre, les objets contenus par ces deux vertex peuvent bouger.

Que peut-on configurer par styles ?

Le tableau ci-dessous va décrire quelques comportements 'customisables' fournis par la bibliothèque.

Comportement	Méthode globale	Propriété
Déplacement	<code>graph.setCellsMovable(false);</code>	<code>mxConstants.STYLE_MOVABLE + "=0"</code>
	<code>graph.setCellsMovable(true);</code>	<code>mxConstants.STYLE_MOVABLE + "=1"</code>
Edition	<code>graph.setCellsEditable(false);</code>	<code>mxConstants.STYLE_EDITABLE + "=0"</code>
	<code>graph.setCellsEditable(true);</code>	<code>mxConstants.STYLE_EDITABLE + "=1"</code>
Redimensionnement	<code>graph.setCellsResizable(true);</code>	<code>mxConstants.STYLE_RESIZABLE + "=0";</code>
	<code>graph.setCellsResizable(false);</code>	<code>mxConstants.STYLE_RESIZABLE + "=1";</code>

Cette liste n'étant pas exhaustive, il faut se référer à la Javadoc de la classe mxConstants pour avoir la liste complète des options possibles.

Pour en finir avec ce chapitre, on peut cumuler plusieurs options comme ceci, chaque propriété étant séparée par des ',' :

```
String style=mxConstants.STYLE_RESIZABLE + "=0;" +
    mxConstants.STYLE_MOVABLE+"=0";
Object level1 = graph.insertVertex(parent, null,
    "Bloc1", 10, 10, 350, 120,style);
Object level2 = graph.insertVertex(parent, null,
    "Bloc2", 10, 150, 350, 120,style);
```

## 5. Comment agir sur l'aspect visuel d'un graphe

JGraphX offre la possibilité de modifier l'aspect visuel d'un graphe en jouant sur les styles des objets dessinés. Comme vu dans le chapitre précédent, il suffit de passer un String lors de la création de l'objet pour en modifier l'aspect.

### 5.1. Aspect visuel des edges

Pour ces éléments on peut modifier le début et la fin d'une flèche.

#### 5.1.1. Modification de la fin d'une flèche

La fin de flèche se modifie avec la propriété mxConstants.STYLE\_ENDARROW. La Javadoc spécifie que l'on peut utiliser les constantes débutant par ARROW, c'est vrai, mais il faut appliquer une limitation supplémentaire : du type String.

Voici quelques exemples :

Propriété :	Aperçu
mxConstants.STYLE_ENDARROW + "=" + mxConstants.ARROW_BLOCK	
mxConstants.STYLE_ENDARROW + "=" + mxConstants.ARROW_CLASSIC	
mxConstants.STYLE_ENDARROW + "=" + mxConstants.ARROW_DIAMOND	
mxConstants.STYLE_ENDARROW + "=" + mxConstants.ARROW_OPEN	
mxConstants.STYLE_ENDARROW + "=" + mxConstants.ARROW_OVAL	

#### 5.1.2. Modification du début d'une flèche

Comme pour la fin d'une flèche, on peut utiliser les mêmes styles pour la constante mxConstants.STYLE\_STARTARROW.

La question qui reste posée pourrait être : mais comment fait-on pour supprimer un bout de flèche? Toutes les constantes débutant par ARROW du type string ont été utilisées, mais aucune ne supprime le bout de flèche.

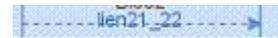
La solution à cette question est :

```
String edgeStyle = mxConstants.STYLE_STARTARROW +
    "=" + mxConstants.NONE;
```

C'est logique, puisque NONE commence bien par ARROW.

### 5.1.3. Le tracé des lignes

Par défaut les lignes des edges sont des lignes pleines. Mais il est possible d'utiliser d'autres types de lignes comme les pointillées. Le plus simple est d'utiliser le style mxConstants.STYLE\_DASHED qui donne le résultat suivant



On peut vouloir aussi modifier l'apparence du pointillé. On le fait en combinant les constantes mxConstants.STYLE\_DASHED et mxConstants.STYLE\_DASH\_PATTERN.

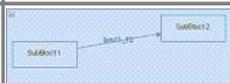
```
String edgeStyle = mxConstants.STYLE_DASHED +
    "=1;" +
    mxConstants.STYLE_DASH_PATTERN + "=10";
Object edge11 = graph.insertEdge(level1,
    null,
    "lien11_12",
    level1_1,
    level1_2,
    edgeStyle);
```

On obtient ce tracé.

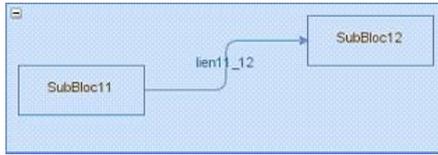


### 5.1.4. La forme des lignes

De base les edges sont représentés par des lignes droites, mais il est possible de modifier cet aspect aussi. Le tableau suivant montre les différentes options possibles.

Propriété	Aperçu
Valeur par défaut	
mxConstants.STYLE_EDGE + "=" + mxConstants.EDGESTYLE_TOPTOBOTTOM;	
mxConstants.STYLE_EDGE + "=" + mxConstants.EDGESTYLE_SIDETOSIDE;	
mxConstants.STYLE_EDGE + "=" + mxConstants.EDGESTYLE_ENTITY_RELATION;	
mxConstants.STYLE_EDGE + "=" + mxConstants.EDGESTYLE_LOOP;	

En associant ces styles avec mxConstants.STYLE\_ROUNDED + "=1" on arrondit alors les angles et cela donne



## 5.2. Aspect visuel des vertex

Comme pour les edges, il est possible aussi de modifier l'aspect visuel des vertex en jouant avec leurs styles.

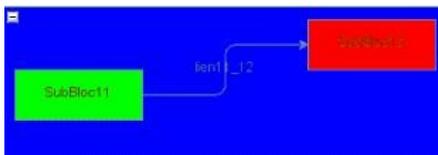
### 5.2.1. Les couleurs des vertex

On peut facilement jouer sur les couleurs de fond et du tour des edges. Prenons l'exemple suivant.

```
String styleParent = mxConstants.STYLE_FILLCOLOR + "#0000ff";
Object level1 = graph.insertVertex(parent, null, "", 10, 10, 350, 120, styleParent);

String styleEnfant1 = mxConstants.STYLE_FILLCOLOR + "#00ff00";
String styleEnfant2 = mxConstants.STYLE_FILLCOLOR + "#ff0000";
Object level1_1 = graph.insertVertex(level1, null, "SubBloc1", 10, 50, 100, 40, styleEnfant1);
Object level1_2 = graph.insertVertex(level1, null, "SubBloc2", 240, 10, 100, 40, styleEnfant2);
```

Donnera le résultat suivant :



Les couleurs de fond des vertex ont donc été modifiées. Le codage de la couleur respecte le principe #RRVVBB ou RR est la valeur hexadécimale du rouge, VV la valeur hexadécimale du vert et BB celle du bleu.

### 5.2.2. Forme des vertex

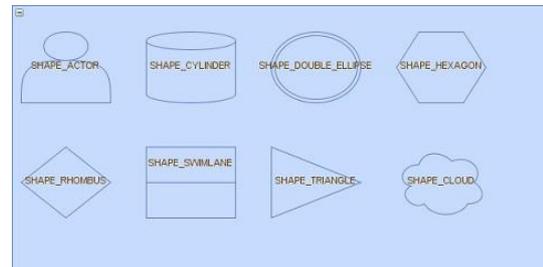
Il est possible aussi de modifier la forme des vertex en jouant avec le style `mxConstants.STYLE_SHAPE` et de lui associer une des constantes commençant par `SHAPE`.

Voici quelques exemples :

```
Object level1 = graph.insertVertex(parent, null, "", 10, 10, 600, 300);
```

```
graph.insertVertex(level1, null, "SHAPE_ACTOR", 10, 30, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_ACTOR);
graph.insertVertex(level1, null, "SHAPE_CYLINDER", 150, 30, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_CYLINDER);
graph.insertVertex(level1, null, "SHAPE_DOUBLE_ELLIPSE", 290, 30, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_DOUBLE_ELLIPSE);
graph.insertVertex(level1, null, "SHAPE_HEXAGON", 430, 30, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_HEXAGON);
graph.insertVertex(level1, null, "SHAPE_RHOMBUS", 10, 160, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_RHOMBUS);
graph.insertVertex(level1, null, "SHAPE_SWIMLANE", 150, 160, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_SWIMLANE);
graph.insertVertex(level1, null, "SHAPE_TRIANGLE", 290, 160, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_TRIANGLE);
graph.insertVertex(level1, null, "SHAPE_CLOUD", 430, 160, 100, 80, mxConstants.STYLE_SHAPE + "="+mxConstants.SHAPE_CLOUD);
```

qui donneront le résultat suivant' :



Je ne listerai pas ici tous les styles que l'on peut utiliser pour modifier l'aspect visuel d'un vertex, je vous laisse le soin de les rechercher dans la Javadoc de la classe `mxConstants`.

## 6. Conclusion

Ce petit tutoriel vous a expliqué les mécanismes de base pour afficher un graphe dans une application Java. à vous de jouer maintenant.

Retrouvez l'article de Patrick Briand en ligne : [Lien 65](#)

### Premiers pas avec Eclipse Scout

Cet article propose une présentation détaillée du framework Eclipse Scout.

Le chapitre II illustre la réalisation complète d'une application à travers un exemple. Le code source de l'exemple est disponible en téléchargement.

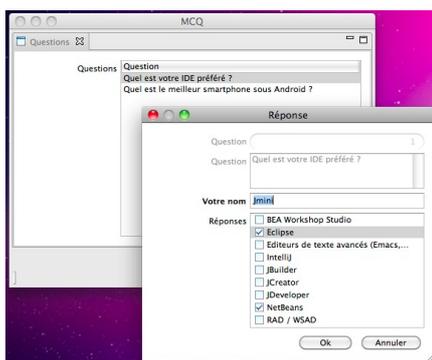
#### 1. Présentation

Eclipse Scout est un outil de création d'applications métier orientées service en Java. Il se situe dans la lignée des ateliers de génie logiciel, en proposant une approche moderne, ouverte et simple d'accès. Il comprend deux composants : une partie SDK (une perspective de plus dans l'IDE Eclipse pour créer l'application) et une partie runtime (intégrée aux applications). Ces deux composants permettent de guider le développeur pour qu'il utilise une architecture définie. C'est l'idée générale d'Eclipse Scout : utiliser les technologies de la plateforme Eclipse pour simplifier et accélérer la réalisation de logiciels.

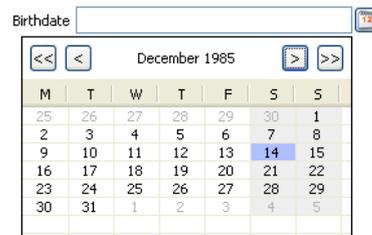
Depuis 2010, c'est un projet de la fondation Eclipse disponible en open source sous licence EPL. En 2011, il est livré pour la première fois avec le *release train* officiel d'Eclipse et est disponible sous forme de *package* officiel sur la page des téléchargements d'Eclipse : [Lien 66](#).

#### 1.1. Applications réalisées

Les applications réalisées avec Eclipse Scout sont robustes, avec une cohérence pour l'utilisateur. De nombreuses fonctionnalités se retrouvent partout. Par exemple, toutes les listes de données peuvent être réorganisées : modification de l'ordre des colonnes par glissé-déposé, affichage/masquage des colonnes, ajout de filtres.



Au niveau des formulaires de saisie des données, l'utilisateur se retrouve en terrain connu. Il retrouve des champs de saisie classiques. De petites fonctionnalités facilitent le travail au quotidien. Par exemple pour le sélecteur de date, l'utilisateur pourra soit utiliser un calendrier, soit utiliser les flèches du clavier pour faire défiler les jours ou encore écrire « +7 » pour fixer la date dans sept jours.



Les applications sont pensées pour être multilingues dès leur création, facilitant le travail de traduction pour disposer d'une application dans la langue de l'utilisateur.

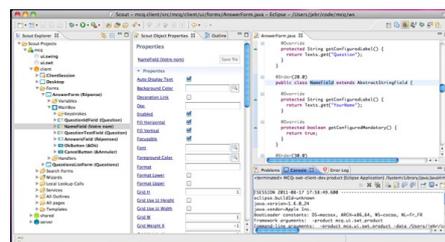
Eclipse Scout s'utilise traditionnellement pour des applications client-serveur, orientées service, utilisant une base de données comme couche de persistance. Comme le code est accessible et qu'il est ouvert, il est possible de réaliser tout autre type d'applications.

Les applications réalisées sont basées sur Java/Eclipse et s'intègrent facilement dans la majorité des environnements IT.

#### 1.2. Partie SDK

Le code des applications Eclipse Scout est du Java lisible. Il définit l'application et sert de référence : il n'y a pas de fichier de configuration ou de cache supplémentaire. Il est possible d'écrire son code « à la main » à l'aide de n'importe quel IDE Java (voire même un simple éditeur de texte).

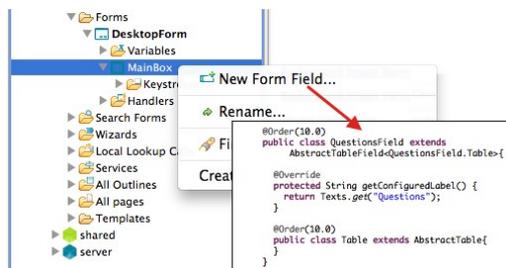
Cependant pour faciliter et accélérer le travail du développeur, Eclipse Scout propose une partie SDK. Il s'agit d'une perspective supplémentaire dans l'IDE Eclipse qui va permettre de générer facilement et de manière intuitive le code Java nécessaire. Ainsi le développeur peut se concentrer sur ce qui compte vraiment pour son application : l'affichage et les fenêtres d'édition des données.



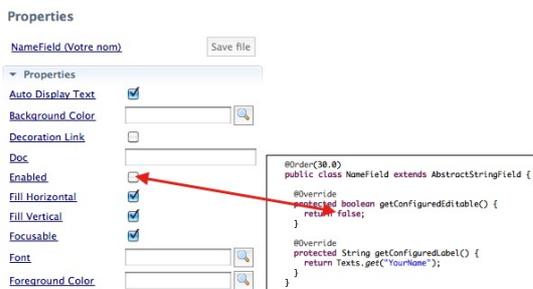
L'idée est de partir de l'interface utilisateur : ce que

l'utilisateur verra. Cette interface influence l'architecture de l'application : services, permissions, objets de transfert de données... Une partie de ce code peut être générée automatiquement, évitant au développeur d'avoir à s'en soucier.

Scout SDK met à disposition plusieurs vues et assistants. La vue *Scout Explorer* représente l'application (sous forme d'arborescence). Différents menus permettent d'agir sur l'application (ajouter un formulaire, un champ...). La vue *Scout Object Properties* va permettre de configurer l'élément actif (choisir le titre d'une fenêtre, l'état par défaut d'un champ). La perspective Scout laisse une grande place à l'éditeur Java : en effet, les vues et assistants créent directement le code Java correspondant. On se situe au même niveau que les outils d'édition du code java proposé par Eclipse (template, refactoring...) avec des possibilités supplémentaires.



Il ne s'agit pas de choisir entre écrire du code à la main (disons dans la perspective Java) ou utiliser la perspective Scout. Les deux approches sont possibles simultanément. Les outils de la perspective créent le code Java, mais ils sont aussi capables de se mettre à jour si le code change par ailleurs. En fonction de ses habitudes et du type de modification à effectuer, il est possible d'utiliser l'une ou l'autre approche.



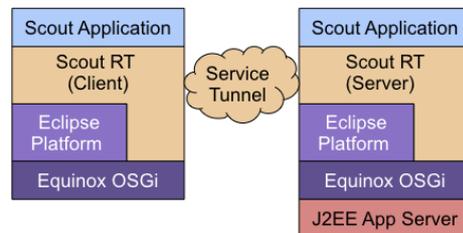
Cette double approche permet de travailler dans des équipes avec un niveau hétérogène. Un développeur qui ne connaîtrait pas le framework va travailler avec la perspective Scout et apprendre en utilisant les outils. Un développeur plus expérimenté pourra faire le choix de coder directement le code Java.

Le code produit par les outils est propre et lisible. Il respecte une architecture définie, sans que le développeur ait à réfléchir. Les outils du SDK créent, modifient et placent le code au bon endroit.

Les applications Eclipse Scout sont pensées pour être fonctionnelles dès leur création à partir d'un template. Lors des phases de prototypage ou de création d'application, cela permet d'accélérer le développement et de pouvoir présenter les modifications rapidement.

### 1.3. Partie Runtime

Le runtime est intégré à chaque application réalisée (partie client ou partie serveur). Il utilise les technologies de la plateforme Eclipse pour créer des applications finies et faire communiquer entre elles de manière transparente la partie client et la partie serveur. Ainsi le développeur se concentre sur l'essentiel : côté client la définition des éléments de l'application vue par l'utilisateur (fenêtres, menus, formulaires...), et côté serveur la gestion des données (celles à saisir ou à afficher).



La partie serveur est hébergée sur un serveur d'application Java (Tomcat, Weblogic...). Elle met à disposition du client un certain nombre de services pour afficher et manipuler les données. L'idée est de s'interconnecter à d'autres services JEE et/ou à une base de données (JDBC). Les services sont définis par des interfaces Java et exposés via OSGi. C'est un système de Service Registry disposant d'un mécanisme de priorité. Les Services peuvent être appelés depuis le client ou le serveur, Eclipse Scout se chargeant de transmettre les données de l'un à l'autre si nécessaire.

La partie client est celle que l'utilisateur final voit. Comme dans toute application, on retrouve les éléments traditionnels : fenêtres, menus, formulaires... Pour la navigation dans les données, Eclipse Scout propose un principe d'arborescence de pages couplé à des formulaires de recherche. D'autres formulaires offrent la possibilité d'éditer les données. En fonction du type d'application, on pourra ouvrir ces formulaires dans des fenêtres modales, ou les présenter successivement (à la manière d'un assistant) pour travailler par processus.

Eclipse Scout propose une stratégie multifront-end. Plusieurs bibliothèques graphiques sont supportées (Swing, SWT, Web avec RAP...). Pour arriver à ce but, seul un modèle du client doit être défini (en Java). Le framework se charge de créer et gérer le rendu de l'interface utilisateur. Ainsi le développeur ne programme pas le détail du rendu, mais se concentre sur la manière dont les données vont être affichées (les interactions et la logique métier), évitant le travail de gestion fine de la bibliothèque graphique.

Pour résumer, la partie runtime s'appuie sur la plateforme Eclipse et simplifie sa mise en œuvre. Cette simplification passe par un certain nombre de choix qui pourraient être limitants pour des besoins spécifiques, mais le framework est suffisamment ouvert pour ajouter les éléments manquants.

### 1.4. Un peu d'histoire

Scout est le framework interne de la société *BSI Business System Integration AG* depuis plus de dix ans. D'abord

basé sur d'autres technologies (notamment des fichiers de configuration XML), c'est un framework 100 % Java depuis 2007.

En 2010, il est soumis à la fondation Eclipse sous forme de *proposal*, ce qui donne naissance au projet Eclipse Scout. Après plusieurs mois pour franchir toutes les étapes du *Eclipse Development Process*, le code source est rendu disponible sur les serveurs de la fondation sous licence EPL vers la fin 2010.

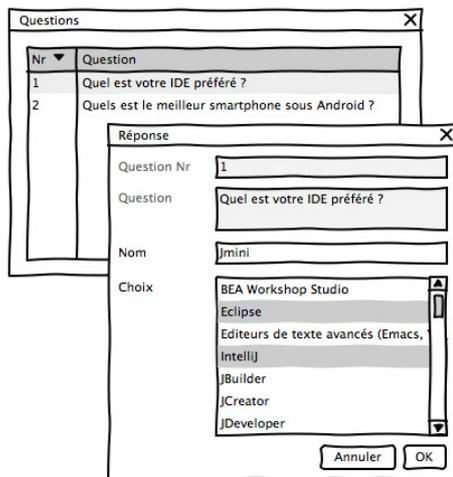
En 2011, Eclipse Scout participe au *release train* annuel d'Eclipse. Avec la version 3.7 (Eclipse Indigo), une distribution packagée est proposée au téléchargement.

## 2. Un exemple complet d'utilisation

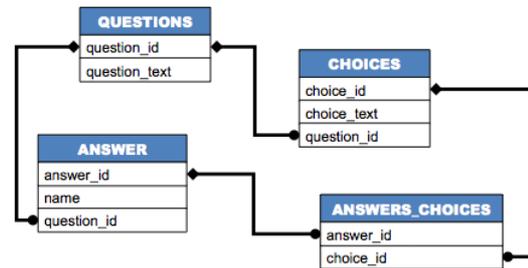
Voici un exemple complet d'utilisation d'Eclipse Scout : de l'installation au déploiement sur un serveur Tomcat. Nous allons créer pas à pas une petite application de QCM (questionnaire à choix multiples).

### 2.1. Aperçu et prérequis

L'application doit afficher la liste des questions et proposer d'ajouter de nouvelles réponses. Une réponse est définie par un nom et une série de choix (on présente une ListBox à l'utilisateur). On déploiera l'application réalisée sur un serveur Tomcat. Voici la maquette de ce à quoi doit ressembler cette application :



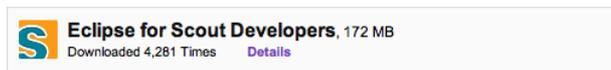
Le code généré par Eclipse Scout est en Java, il faut donc en posséder les rudiments. Comme les données sont stockées dans une base de données, il est préférable de connaître un minimum le SQL. Ce tutoriel utilise principalement les outils du SDK, il peut donc s'adresser à des développeurs débutants.



La base de données Derby doit déjà être installée et contenir quelques questions. Cette base est rudimentaire, mais contient le nécessaire pour l'application de QCM : une table pour les questions, reliée avec des choix et des réponses. Chaque réponse peut contenir plusieurs choix. Un script de création de la base et de remplissage de données est fourni en annexe (voir V-A). Sans base de données il faudra ignorer l'étape III-5 et certaines parties de l'étape III-6. Dans ce cas, l'application sera fonctionnelle, mais ne gardera pas mémoire des modifications.

### 2.2. Installation

Le moyen le plus simple pour obtenir une version d'Eclipse avec les outils d'Eclipse Scout est de télécharger la version dédiée « *Eclipse for Scout developers* » sur la page des téléchargements d'Eclipse : [Lien 66](#).



Pour les connaisseurs d'Eclipse, il est également possible d'utiliser l'update site d'Eclipse Indigo. [Lien 67](#). Dans ce cas, il faudra ajouter la perspective Scout manuellement après avoir relancé Eclipse.

Retrouvez la suite de l'article de Jérémie Bresson en ligne : [Lien 68](#).



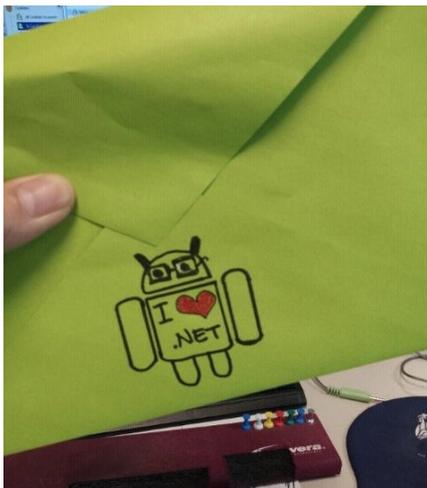
### Java remplacé par C# dans Android ?

**Xamarin développe XobotOS, une solution qui annule l'utilisation de Java dans l'OS mobile de Google**

Alors que le procès entre Oracle et Google pour violation de brevet Java dans Android bat son plein ([Lien 69](#)), la startup Xamarin développe une solution pouvant permettre de ne plus utiliser le langage dans l'OS.

La société à l'origine de Mono pour Android, l'implémentation open source du framework .NET permettant la conception des applications natives Android en C#, s'est lancée dans un projet ambitieux avec pour objectif de remplacer le code Java dans Android par du code C#.

Le projet a donné naissance à XobotOS, une plateforme d'exécution d'applications pour Android entièrement en C# qui ne nécessite pas Java.



Pour parvenir à ce résultat, les développeurs de Xamarin ont utilisé Sharpen, un outil de conversion de code Java en C#. La société a dû améliorer Sharpen afin qu'il puisse gérer la complexité de la base du code Android. Certaines portions du code Java ont dû être portées manuellement.

L'analyse des performances de XobotOS a permis de constater une exécution plus rapide des programmes .NET par rapport aux programmes Java équivalents dans Android. Selon les développeurs du projet, la machine virtuelle Mono est mature avec des optimisations plus poussées que la machine virtuelle Dalvik de Google et profite de certaines fonctionnalités du Framework.NET dont ne dispose pas Java.

De plus, cette solution serait même plus avantageuse pour Google dans la mesure où Microsoft a proposé le langage C# et la machine virtuelle .NET à l'ECMA pour standardisation, et la plateforme est couverte par la licence Microsoft Community Promise license permettant à

Google d'obtenir une licence peu coûteuse.

Pour l'instant, XobotOS n'est qu'un projet de recherche pour Xamarin qui n'a pas l'intention de le maintenir comme un projet autonome, mais l'utiliser comme une couche d'accès direct au système d'exploitation plutôt que de passer par la machine virtuelle Java pour améliorer Mono pour Android.

XobotOS a été publié comme un projet open source sous les termes de la licence Apache, sur GitHub. Une solution sur laquelle devrait se pencher Google pour contourner l'utilisation de Java dans Android ?

**Le projet sur GitHub :** [Lien 70](#)

*Commentez cette news de Hinault Romaric en ligne : [Lien 71](#)*

### Google Drive : le nouveau service de stockage de Google est ouvert

**Utilisé avec Chrome, il permet l'envoi de fax ou la création de maquettes de sites**



On s'y attendait ([Lien 72](#)), c'est confirmé. Google a annoncé hier soir le lancement officiel de Google Drive, un nouvel espace de stockage en ligne pour créer, partager, collaborer et conserver des documents.

« Vous pouvez télécharger et accéder à tous vos fichiers, des vidéos aux PDF, en passant par vos photos et Google Documents », explique Google.

Quelle différence avec Google Documents justement, qui est déjà compatible avec tous ces formats de fichiers ?

« **Google Documents est directement intégré à Google Drive** », répond Google France. Drive en hérite des fonctionnalités comme la reconnaissance de caractères (« si vous téléchargez l'image numérisée d'une vieille coupure de journal, vous pouvez rechercher l'un des mots cités dans l'article ») ou la modification en ligne et à plusieurs des textes, présentations et des tableaux.

« Nous avons même commencé à exploiter la reconnaissance d'images », se félicite l'éditeur. « Si vous glissez-déposez une image de la Tour Eiffel dans Drive, la prochaine fois que vous rechercherez le terme [Tour Eiffel], cette image apparaîtra dans les résultats ». Une fonctionnalité qui reste encore expérimentale.

Mais Google Drive va plus loin. Le service peut ouvrir **plus de 30 types de documents** directement depuis un

navigateur Web - dont des vidéos haute-définition, Adobe Illustrator, Adobe Photoshop.

La principale nouveauté de Drive est en fait ailleurs. Pour concurrencer Dropbox et cie, le service propose **une application multiplateforme** (aujourd'hui Mac, Windows, Android et bientôt sur iOS).



Il **s'intègre également aux autres produits Google** et permet par exemple de partager des photos depuis Drive sur Google+ ou – bientôt – de joindre des documents de cet espace de stockage directement dans des mails envoyés avec Gmail.

Utilisé **avec le navigateur maison Chrome**, Google Drive offre encore d'autres possibilités via des extensions dédiées : **l'envoi de fax, l'édition de vidéos ou la création de maquettes de sites** Internet (le tout directement depuis le service).

« Et ce n'est que le début, de nombreux développements sont à venir », promet Google.

[Lien 73](#)

Tout comme les Google Docs et Ubuntu One, le service de Canonical, Google Drive propose 5 Go de stockage gratuit (contre 7 Go pour SkyDrive de Microsoft ([Lien 74](#)), qui vient de baisser ce plafond mais qui dans le même temps a lui aussi publié des applications pour Windows et Mac).

Côté professionnels, Google propose de **nouveaux outils dans l'interface de contrôle des Google Apps for Business** pour les administrateurs (afin d'ajouter ou de supprimer de l'espace de stockage pour des utilisateurs individuels ou groupés), le **chiffrement du transfert de données** (mais toujours **pas de chiffrement asymétrique pour le stockage lui-même** ([Lien 75](#))), la vérification en deux temps (via un code envoyé sur mobile), et une **disponibilité de 99,9 % garantie**.

L'espace de **stockage est bien évidemment extensible**. Les administrateurs peuvent le gérer et l'étendre de manière centralisée. Lorsqu'un utilisateur atteint la limite, les administrateurs peuvent acheter 20 GB supplémentaires. Petite incertitude néanmoins, la page du service affiche un prix inférieur à 2,5 \$ par mois pour cet ajout quand le blog français indique lui 4 \$ par mois (les prix en euros ne sont visiblement pas encore fixés).

Après l'annonce du lancement de Google Drive, **deux critiques sont tombées** sur le service.

La première concerne **les conditions générales d'utilisation**.

Certains y voyaient la possibilité pour Google de s'emparer des données hébergées sur Drive. Une accusation **que Google a immédiatement démentie** quand nous les avons contactés : [Lien 76](#).

La **deuxième critique** concernait la dimension multiplateforme de l'outil. L'application Google Drive n'étant **pas disponible sur Linux**.

Ajouté à la récente annonce de l'abandon d'un produit populaire comme Picassa dans sa version Linux, il n'en fallait pas plus pour créer de la défiance (comme le montre ce commentaire, ici même : [Lien 77](#)).

Que les utilisateurs de l'OS open source se rassurent, Teresa Wu, une porte-parole de la société, a confirmé hier soir dans le cadre d'une discussion sur Google+ que **le travail était en cours pour porter l'application sur la plate-forme**.

Un Google Drive pour Linux est donc bien prévu. Reste à savoir pour quand.

**Google Drive et ses applications locales sont lancés sur cette page** : [Lien 78](#)

Commentez cette news de Gordon Fowler en ligne : [Lien 79](#)

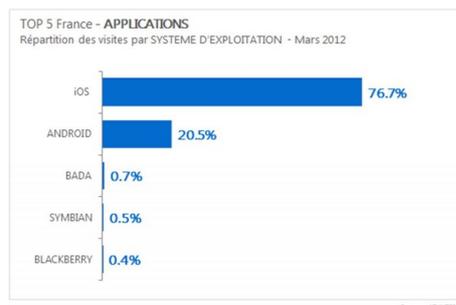
## « Android est plus dynamique en France qu'iOS »

**D'après AT Internet, qui publie les résultats d'une étude qui pose plusieurs questions**

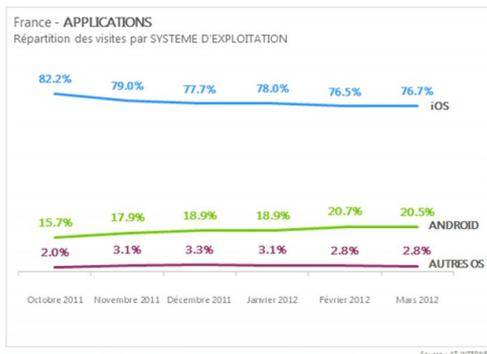
AT Internet vient de publier une nouvelle étude sur les parts de marché des OS mobiles. Cette étude s'appuie sur « une estimation de la répartition des visites par systèmes d'exploitation pour 337 applications, tous secteurs confondus (applications auditées par AT Internet) ». Pour le dire simplement, en s'appuyant sur le trafic Web généré par ces applications (phénomène dit « d'appification » : [Lien 80](#)).

D'après le cabinet d'analyse, la conclusion serait claire : « *Android est plus dynamique en France qu'iOS* ».

Pourtant, Apple reste le leader incontesté du marché avec 76,7 % des visites françaises, très loin devant Android.

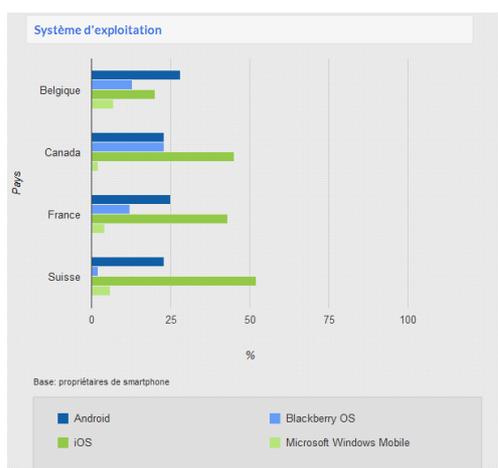


Côté progression, il est vrai qu'iOS perd 5.5 points en 6 mois.



Mais à ce rythme, il faudra encore 28 mois pour qu'Android arrive au niveau du système d'exploitation d'Apple. Le dynamisme de l'OS de Google sera donc relativisé par certains.

AT Internet évalue les parts de marché françaises des autres concurrents à 0,7 % des visites pour Bada, à 0,5 % pour Symbian et à 0,4 % pour Blackberry. Windows Phone n'est pas présent dans ces résultats. Windows Phone qui bien qu'ayant une faible pénétration du marché n'en est cependant pas totalement absent comme l'a récemment montré Google dans une étude remarquable sur les Smartphones dans le monde : [Lien 81](#).



« *Our Mobile Planet* » apportera d'ailleurs de très nombreux éléments complémentaires aux développeurs et aux décideurs IT.

Il apporte aussi un éclairage particulier sur les estimations d'AT Internet, qui se cantonnent aux applications elles-mêmes et pas au surf mobile dans son ensemble, et il montre qu'elles ne sont pas sans poser de nombreuses questions.

### Et vous ?

D'après vous, Android est-il plus dynamique qu'iOS en France ou faut-il relativiser cette vision ?

Que pensez-vous de l'absence de Windows Phone : logique ou preuve que AT Internet n'a pas inclus d'applications de cet OS dans son panel ?

D'après Google, BlackBerry possède 12 % du marché français. Comment expliquer qu'AT Internet ne lui attribue « que » 0,4 % du trafic Web issu d'applications ?

Et plus largement, peut-on évaluer de manière pertinente des parts de marché en s'appuyant uniquement sur les trafics générés par des applications ?

Commentez cette news de Gordon Fowler en ligne : [Lien 82](#)

# Business Intelligence

## Les derniers tutoriels et articles



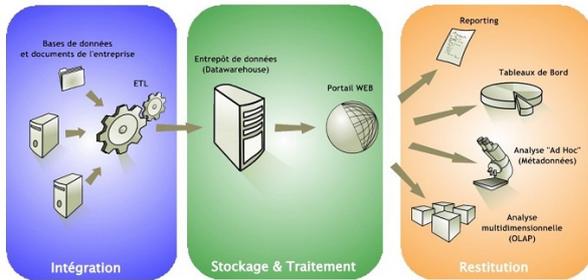
### Présentation et installation de Pentaho

Ce tutoriel, destiné aux débutants, a pour objectif de vous présenter la plate-forme Pentaho.

#### 1. Présentation de Pentaho

Pentaho est une plate-forme décisionnelle open source complète possédant les caractéristiques suivantes :

- une couverture globale des fonctionnalités de la Business Intelligence :
  - ETL (intégration de données),
  - reporting,
  - tableaux de bord ("Dashboards"),
  - analyse *ad hoc* (requêtes à la demande),
  - analyse multidimensionnelle (OLAP) ;

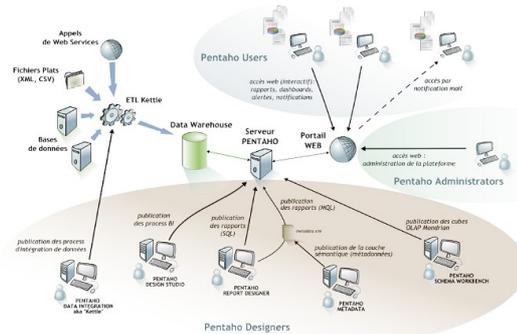


- Pentaho permet d'adresser deux typologies d'utilisateurs :
  - les « one-clic users », utilisateurs de base, consommateurs d'indicateurs prédéfinis,
  - les utilisateurs avancés, qui ont besoin d'outils d'analyse et d'exploration avancés ;



- une architecture Web 2.0 qui se compose :
  - d'un serveur Web J2EE permettant de mettre à disposition l'ensemble des ressources décisionnelles et ceci au travers d'URL Web uniques et standardisées. Le serveur est dénommé "Pentaho User Console" (PUC),
  - plusieurs clients riches permettant la conception et la publication des ressources. Ces derniers sont librement téléchargeables et peuvent être installés

sous des environnements Windows, Linux ou Mac-OS (clients Java) ;



- le serveur Web Pentaho comporte également une plate-forme d'administration (*Pentaho Administration Console*) pour la gestion des droits d'accès, la planification d'événements, la gestion centralisée des sources de données... ;
- Pentaho est reconnue pour être une solution d'une grande qualité conceptuelle et technique. La plate-forme est orientée « processus » : au travers de « séquences d'actions » on peut ainsi modéliser avec Pentaho des workflows BI avancés ;
- il n'est pas besoin de connaître JAVA pour travailler avec Pentaho : seule la maîtrise du langage SQL est nécessaire, ainsi que des connaissances de base en XML, HTML et JavaScript. Il faut bien sûr s'autoformer (ou être formé) aux clients de conception ;
- une communauté ([Lien 83](#)) importante et très active s'anime autour de Pentaho. Celle-ci contribue au codage de nombreux plugins et de projets communautaires : plugins Kettle, Pentaho Analysis Tool, Pentaho Community Dashboard Framework, etc. ;
- Pentaho est une suite décisionnelle open source commerciale qui reste très « ouverte ». Les différences fonctionnelles entre la version libre (*community edition*) et la version payante (*enterprise edition*) restent limitées. **La version libre de Pentaho permet d'installer une plate-forme décisionnelle complète !**

#### 2. Téléchargement de Pentaho

Pour débiter avec Pentaho, il est conseillé de télécharger la version community, gratuite et libre d'utilisation. Cette version communautaire peut-être téléchargée sur SourceForge ([Lien 83](#)) ici : [Lien 84](#).

On y retrouve le serveur Pentaho ("*Business Intelligence Server*") ainsi que tous les clients de conception (voir détail au paragraphe suivant) :

File/Folder Name	Platform	Size	Date	Downloads	Notes/Subscribe
Newest Files					
<a href="#">pme-ce-mac-3.5.0-stable.zip</a>		48.9 MB	2010-03-18	657	
All Files					
▶ Pentaho Metadata	PME	587.1 MB	2010-03-18	34,345	
▶ Report Designer	PRD	1.1 GB	2010-03-04	63,421	
▶ Business Intelligence Server	BISERVER	7.3 GB	2010-02-25	497,751	
▶ Data Integration	PDI (Kettle)	1.4 GB	2009-12-21	147,210	
▶ Design Studio	PDS	3.4 GB	2009-10-15	104,531	
▶ Report Design Wizard (Legacy) - (obsoleto)		181.7 MB	2008-08-29	50,102	
▶ White Papers (documents "anciens"...		1.6 MB	2006-06-20	57,355	

**Les numéros des versions téléchargées pour les clients de conception et le serveur Pentaho (biserver) doivent toujours être en adéquation : par exemple Pentaho Report Designer 3.0.0 avec Biserver 3.0.0.**

Des paquetages d'installation sont disponibles pour tous les systèmes d'exploitation :

- Windows ;
- Linux ;
- Mac.

Report Designer		1.1 GB	2010-03-04	63,421	
▶ 3.0.0-stable		184.5 MB	2010-03-04	4,174	
▶ prd-source-3.6.0-stable.zip	Code Source	591.0 KB	2010-03-04	634	
▶ prd-ce-mac-3.6.0-stable.tar.gz	MAC	54.6 MB	2010-03-04	335	
▶ prd-ce-3.6.0-stable.zip	WINDOWS	54.7 MB	2010-03-04	2,489	
▶ prd-ce-3.6.0-stable.tar.gz	LINUX	54.6 MB	2010-03-04	716	

### 3. Liste des clients de conception Pentaho

Outil	Code	Fonction
<b>Pentaho Report Designer</b>	<b>PRD</b>	Client de conception de rapports avancés. Il s'agit d'un outil de mise en page similaire à iReport, Eclipse BIRT, Crystal Reports... Permet de se connecter à de nombreuses sources de données : SGBD, XML, Excel, CSV, flux de données venant de Kettle, MDX (OLAP)...
<b>Pentaho Design Studio</b>	<b>PDS</b>	Client Eclipse de modélisation de workflows BI (Xactions) propre à Pentaho. Design Studio permet de mettre en œuvre de nombreuses ressources BI en minimisant l'écriture de code (envoi de mails automatisé par ex.).
<b>Pentaho Metadata</b>	<b>PME</b>	Client riche permettant la mise en place d'une couche sémantique d'abstraction (métadonnées) sur la couche physique (tables et colonnes d'une base de données). Le but est de rendre les objets d'un SGBD compréhensibles et manipulables par un utilisateur

		final afin de lui permettre d'effectuer ses propres requêtes et ceci sans connaître le langage SQL. La couche de métadonnées peut-être utilisée dans le requêteur Web <i>ad hoc</i> , dans Pentaho Report Designer et dans Pentaho Design Studio.
<b>Pentaho Schema Workbench</b>	<b>PSW</b>	Client riche permettant la définition des schémas Mondrian à partir d'un modèle en étoile ou flocon de l'entrepôt de données. Un autre outil, Pentaho Aggregation Designer (PAD), permet de construire et de charger automatiquement des tables d'agrégation en vue d'améliorer les performances lors du requêtage des cubes Mondrian. <b>Téléchargement spécifique ici : <a href="#">Lien 85</a></b>
<b>Pentaho Data Integration (Kettle)</b>	<b>PDI</b>	Outil ETL (Extract Transform Load) complet, pouvant être utilisé indépendamment de la plate-forme Pentaho. Kettle est comparable à Talend Open Studio en termes de fonctionnalités. Pour consulter les différences techniques et fonctionnelles, un livre blanc est disponible à cette adresse : <a href="#">Lien 86</a>

### 4. Installation de Pentaho (en local)

Le serveur Pentaho (*biserver-ce*) est un serveur de démonstration prêt à l'emploi, complètement autonome et pouvant être installé sur un PC bureautique disposant au moins de 1 Go de RAM.

Ce serveur s'appuie notamment sur le système de gestion de base de données Hypersonic (HsqlDb) pour le stockage des données exemples (*SampleData*) ainsi que des deux bases internes de Pentaho (*hibernate* et *quartz*).

**HsqlDb étant un système de base de données gérée en mémoire, il est fortement déconseillé de déployer cette configuration en production !**

Une fois l'archive téléchargée, il suffit de décompresser celle-ci dans un répertoire préalablement créé, par exemple « C:\Pentaho-3.5.2 » (Windows).

Le répertoire d'installation sera désigné {PENTAHO-HOME} dans la suite de ce document.

Deux répertoires sont créés dans {PENTAHO-HOME} :

- **\biserver-ce** : la console Web d'utilisation (Pentaho User Console) ;
- **\administration-console** : la console Web pour l'administration de la plate-forme (Pentaho Administration Console).

## 5. Démarrer & arrêter les serveurs Pentaho

### 5.1. Pentaho User Console

Les commandes suivantes permettent de lancer et stopper la console d'utilisation Web

Action	Commande
Démarrage (Windows)	{PENTAHO-HOME}\biserver-ce\start-pentaho.bat
Arrêt (Windows)	{PENTAHO-HOME}\biserver-ce\stop-pentaho.bat
Démarrage (Linux)	{PENTAHO-HOME}\biserver-ce\start-pentaho.sh
Arrêt (Linux)	{PENTAHO-HOME}\biserver-ce\stop-pentaho.sh

On accède à la console d'utilisation Pentaho en saisissant l'URL suivante dans un navigateur Web :

***http://localhost:8080/pentaho***

Puis en saisissant l'identifiant et mot de passe ci-dessous :

- **login** : joe ;
- **password** : password.

### 5.2. Pentaho Administration Console

Les commandes suivantes permettent de lancer et stopper la console d'utilisation Web

Action	Commande
Démarrage (Windows)	{PENTAHO-HOME}\administration-console\start-pac.bat
Arrêt (Windows)	{PENTAHO-HOME}\administration-console\stop-pac.bat
Démarrage (Linux)	{PENTAHO-HOME}\administration-console\start-pac.sh
Arrêt (Linux)	{PENTAHO-HOME}\administration-console\stop-pac.sh

On accède à la console d'administration Pentaho en saisissant l'URL suivante dans un navigateur Web :

***http://localhost:8099***

Puis en saisissant l'identifiant et mot de passe ci-dessous :

- **login** : admin ;
- **password** : password.

*Retrouvez l'article de Sylvain en ligne : [Lien 87](#)*

## De SAS Foundation à SAS BI : Comment créer son Dashboard KPI

L'article détaille la mise en place d'un Dashboard KPI grâce aux trois outils SAS : SAS Base, SAS Enterprise Guide et SAS BI Dashboard. Ainsi selon la plate-forme de travail, le lecteur pourra concevoir des tableaux de bord synthétiques ayant un fort impact visuel et diffuser l'information au sein de l'entreprise.

### 1. Introduction

L'article propose la mise en place des Dashboard KPI SAS au travers de trois outils de l'éditeur éponyme. Le terme «Dashboard» est souvent employé dans le monde du décisionnel pour parler de tableaux de synthèse présentant les indicateurs-clefs de l'activité (connus sous le sigle anglais KPI - *Key Performance Indicator*). Les Dashboard ont une réelle valeur ajoutée pour un travail de reporting en raison de leur aspect visuel et leur simplicité de mise en oeuvre.

Même si la désignation «tableaux de bord» répond à ce besoin, elle apparaît quelque peu galvaudée et regroupe désormais beaucoup de choses (tableaux volumineux, graphiques de toutes sortes) ce qui l'éloigne d'une notion de synthèse, très courte mais surtout visuelle, d'une activité telle que nous le verrons ici.

Nous poursuivrons donc la discussion en désignant les tableaux de présentation des KPI par «Dashboard» ou «Dashboard KPI».

Après une présentation générale des KPI, nous parlerons de la procédure de base fournie avec SAS 9.2 permettant de construire un graphique KPI sans utiliser de notions de métadonnées et de reporting Web. Il est conseillé de jeter

un oeil à cette partie avant d'aborder les deux autres.

Dans une seconde partie, nous emploierons de nouveau cette procédure, mais au travers de SAS Enterprise Guide 4.3 et de l'add-in dédié à cette tâche. Cette partie permet de faire le pont entre SAS Foundation et le travail dans un environnement BI qui s'illustre par l'utilisation des métadonnées et des outils comme Web Report Studio.

Enfin, une troisième partie conclura la présentation avec la mise en place de ce même Dashboard grâce à l'outil SAS BI DASHBOARD accessible par une URL dans un environnement exclusivement BI.

Les exemples et illustrations ont été réalisés sur une plate-forme Windows 32 bits avec :

- SAS EBI 9.2 M3 ;
- SAS BI Dashboard 4.3 ;
- SAS Enterprise Guide 4.3 ;
- L'add-in Office 4.3.

Les mêmes résultats sont obtenus sur une plate-forme Unix et Windows 64 bits.

Les données employées sont celles de la table SASHELP.CARS, regroupant les valeurs de ventes et de

consommation d'essence pour 488 modèles de voitures réparties selon leur fabricant et l'origine de fabrication.

## 2. Principe général des Dashboard KPI

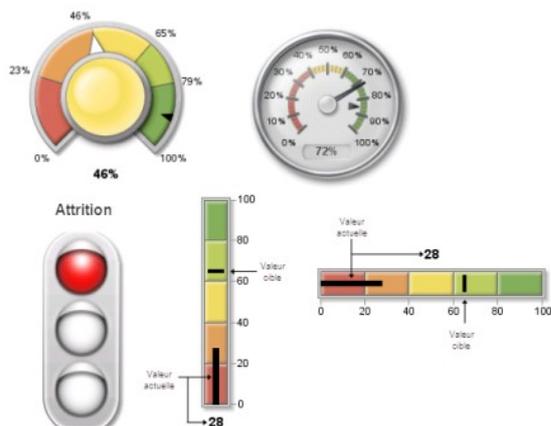
Un Dashboard KPI a pour objectif de présenter sur une page, les indicateurs-clefs de suivi de l'activité d'une société. On souhaitera par exemple rapidement voir où se situe le chiffre d'affaires du mois par rapport aux objectifs, le volume de vente par rapport à des seuils de performances, etc.

L'intérêt est d'avoir un visuel fort permettant de comprendre rapidement l'activité.

Pour cela, la plupart des outils du commerce proposent des éléments tels que des jauges, des accélérateurs ou des feux tricolores. SAS n'est pas en reste et a développé le même principe pour sa plate-forme BI, mais également pour les utilisateurs de plate-forme FOUNDATION, qui souhaitent donner un impact visuel fort à leur reporting.

### 2.1. Les indicateurs

Voici les objets que l'on peut manipuler à cet effet :



L'objectif est donc de visualiser un indicateur d'activité. Chaque indicateur, que nous nommerons souvent KPI, pourra être présenté avec une image telle une jauge ou un cadran. À cela s'ajoutent des seuils et des objectifs qui se présenteront sous la forme de barres, de flèches ou de segments de couleur.

### 2.2. Les éléments constitutifs d'un indicateur

Dans l'article, nous parlerons de valeur actuelle, seuils et valeur cible. Ce sont ces trois notions qui sont essentielles aux Dashboard :

1. La valeur actuelle est la valeur à suivre pour un indicateur donné. Par exemple, le chiffre d'affaires du mois est la valeur actuelle ;
2. On pose ensuite des seuils permettant de mesurer dans quel segment, la valeur actuelle se situe ;
3. Enfin la valeur cible sert à afficher un objectif.

Les couleurs sont personnalisables comme nous le verrons ainsi que l'apparition de symboles pour les valeurs actuelles et cibles telles que des aiguilles dans les compteurs ou des marques sur les jauges.

## 3. Développer son Dashboard avec SAS Foundation

Ce chapitre montre que les Dashboard peuvent être développés et diffusés sans plate-forme BI SAS. On retrouve ici le fonctionnement traditionnel de SAS dans lequel le programmeur génère un document Web ou Office et le met à disposition sur le réseau ou le pousse par mail vers les décideurs.

Ce chapitre permet également d'introduire plus précisément les représentations des indicateurs que l'on retrouvera plus tard au travers de SAS Enterprise Guide du SAS BI Dashboard.

### 3.1. Procédure SAS associée à la génération des KPI

La procédure sous SAS permettant de développer les Dashboard est la proc GKPI.

Elle permet de définir le type de KPI comme présenté dans le tableau suivant puis de définir la qualité du graphique. Ainsi, selon le souhait de présentation, un indicateur-clef prendra telle ou telle forme.

Désignation dans la procédure	Description
dial	Le cadran présente la valeur actuelle et des seuils. Le centre du cadran est de la même couleur que le segment qui contient la valeur réelle.
Speedometer	Des pointeurs indiquent des seuils. Des niveaux de couleurs viennent ajouter une autre information. L'accéléromètre peut être plein, en demi-cercle ou un quart de cercle.
Vtrafflight/htrafflight	Contient une lumière par segment de valeur. Le segment contenant la valeur actuelle est en couleur. Le fait que les valeurs limites ne sont pas affichées peut fausser la présentation.
Vslider/hslider ; vbullet/hbullet	Barre séparée par des seuils. La valeur est indiquée par un niveau (bullet) ou une flèche (slider)

Elle prend ensuite en paramètre les valeurs de l'indicateur et les bornes délimitant les segments de valeurs.

### 3.2. Renseigner les valeurs à visualiser et les seuils

La procédure accepte des valeurs entrées directement avec le code et ne sait pas lire une table.

#### 3.2.1. Une première approche avec une saisie manuelle des KPI

La procédure GKPI ne lit pas directement une table comme le font les autres procédures. Il s'agit ici de renseigner directement la valeur actuelle, la valeur cible et les seuils.

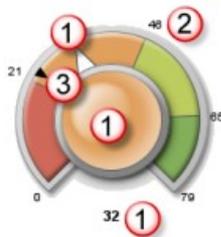
Par exemple, si l'on souhaite un cadran avec l'affichage du chiffre d'affaires du mois, de l'objectif et de quatre seuils, il faut fournir six valeurs manuellement ou par des

macros-variables.

Ainsi, nous aurons à mettre :

```
proc gkpi mode=raised;
dial actual=32
bounds=(0 21 46 65 79)
/ target=23
```

Pour afficher ceci :



On y voit :

1. La valeur actuelle est paramétrée avec actual=32. Elle est symbolisée avec une flèche au sein des segments ; il faut noter également que la couleur du cadran central change en fonction du segment dans lequel se trouve la valeur actuelle ;
2. Les seuils des segments valorisant la valeur actuelle, paramétrée avec bounds=(0 21 46 65 79) ;
3. La cible, symbolisée par la flèche noire plus petite que la flèche de la valeur actuelle, est paramétrée avec target=23 ;

Pour le moment, cette procédure ne peut pas lire une table, ce qui représente une limite que SAS devra combler.

### 3.2.2. Récupérer les valeurs d'une table pour visualiser les KPI

Un programmeur SAS comprendra rapidement que si des valeurs doivent être reprises d'une table, elles le seront au travers de macros-variables pour contourner cette limitation. La valeur cible devrait quant à elle provenir des directions métiers, car il est rare (mais possible) de trouver les objectifs de la société dans une base de données. Enfin pour les seuils, soit ils sont renseignés en accord avec les métiers, soit le statisticien déterminera des seuils grâce aux quartiles par exemple.

Pour illustrer cela, l'exemple suivant va calculer la valeur moyenne des ventes des voitures américaines depuis la table SASHELP.CARS. Les seuils seront déterminés dynamiquement avec des mesures statistiques liées au montant des factures.

Avec le code suivant commenté, un graphique est alors généré :

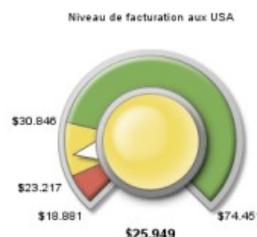
```
proc summary data=sashelp.cars; /*Une proc
Summary permet de calculer des indicateurs
statistiques de ventes */
where origin = "USA" ; /*Restriction aux États-
Unis */
var invoice;
output out=cars mean=v_act Q1=s1 median=s2 Q3=s3
max=s4; /*Calcul de la moyenne et des
quartiles et de la valeur maximale */
```

```
run;

data _null_;
set cars; /*Les statistiques sont
enregistrées dans les macros-variables */
call symputx ( '_act' , v_act); /*La
valeur actuelle est dans &_act */
call symputx ( '_s1' , s1); /*Le premier
quartile est dans &_s1 */
call symputx ( '_s2' , s2); /*La médiane est
dans &_s2 */
call symputx ( '_s3' , s3); /*Le troisième
quartile est dans &_s3 */
call symputx ( '_s4' , s4); /*La valeur
maximale est dans &_s4 */
run;

proc gkpi mode=raised;
dial actual=&_act /*Les macros-variables
sont exploitées */
bounds=(&_s1 &_s2 &_s3 &_s4 ) /
format="dollarx8.0" /*Un format Dollar est
appliqué aux valeurs */
label="Niveau de facturation aux USA"; /*Un
libellé est créé sur le KPI */
run;
quit;
```

Le résultat est celui-ci.



La valeur actuelle de 25 949 \$ se retrouve dans le second segment de valeurs ce qui amène une couleur centrale jaune.

Les ventes moyennes sont supérieures à la médiane ce qui est plutôt positif (le nombre de voitures chères se vendant plutôt plus que les voitures peu onéreuses). On constate par contre qu'il y a un énorme écart entre le seuil du troisième quartile et le prix de la voiture le plus élevé.

### 3.3. Les options à prendre en compte lors de la conception du programme

Les options suivantes permettent d'améliorer le rendu des graphiques.

#### 3.3.1. Moteur de génération

Tout d'abord, l'image générée par la procédure ne peut qu'exploiter un moteur JAVAIMG. On posera donc en premier lieu :

```
Goptions device = javaimg ;
```

Ce moteur graphique permet de générer des images au format Portable Network Graphics (PNG). Ce format assez récent remplace le format GIF assez avantageusement lorsqu'il s'agit de diffuser des images sur le Web avec une palette de couleurs étendue et un fond transparent.

Il est possible dans ce cas d'utiliser un ODS HTML ou ODS RTF pour regrouper différents indicateurs. Chacun d'entre eux allant générer une image en plus de la page Web.

Ainsi le code suivant génère un fichier KPItest.html pour une sortie. Les images seront stockées dans F:\temp car le PATH est spécifié, autrement le fichier HTML et les images iraient dans C:\Program Files\SAS\SAS Foundation\9.2 (là où le moteur SAS est installé).

```
ods html path="F:\temp" (url=none)
file='KPItest.html' style=listing;
```

### 3.3.2. Profondeur du graphique

La procédure permet de définir si l'objet sera en deux ou trois dimensions. La valeur RAISED permet de créer un graphique en trois dimensions :

```
Proc GKPI mode=raised ;
```

### 3.3.3. Couleurs des segments

Les couleurs sont par défaut les suivantes : rouge, orange, jaune, jaune-vert, vert. Si quatre segments sont utiles seulement, le jaune est supprimé.

Couleur RGB	Valeur hexadécimale
Red	cxD06959
Orange	cxE1A05D
Yellow	cxF1DC63
Yellow-Green	cxBDCD5F
Green	cx84AF5B
Grey	cxb2b2b2

Les jeux de couleurs SAS/GRAPH sont de toute façon supportés par la procédure.

L'étape COLORS=() permet de spécifier l'ordre des couleurs utilisées. En spécifiant la couleur GREY, le graphique sera en noir et blanc.

L'étape ACTIVECOLORS = () permet de forcer la couleur du segment dans lequel se trouve la valeur actuelle. Elle n'a pas besoin de contenir une couleur par segment. Cette étape n'est pas souvent utilisée.

### 3.3.4. Personnalisation de l'affichage

Le tableau suivant résume les possibilités offertes pour modifier l'apparence du graphique. Il faut prendre garde à éviter les valeurs décimales, car cela peut alourdir la lecture.

Option	Description
AFONT=	Police de la valeur actuelle
AVALUE NOAVALUE	Affiche ou non la valeur actuelle
BFONT=	Police pour les seuils
BVALUE	Affiche ou non les seuils

NOBVALUE	
FORMAT	Format SAS donné aux valeurs
LABEL=	Libellé à afficher en haut du graphe (256 caractères)
LFONT=	Police du libellé en haut du graphe
LOWBOUND  LOWBOUNDARY	Indique comment la couleur de la valeur actuelle doit être affichée lorsqu'elle coïncide avec le seuil. La couleur du segment précédent est utilisée avec la valeur LOWBOUND
NAME=	Nom du fichier généré. Par défaut c'est graph.png
TARGET=	Valeur cible pouvant être comparée à la valeur actuelle

Parmi les options générales applicables aux graphiques SAS, certaines fonctionnent avec la proc GKPI

Option	Description
TITLE, FOOTNOTE	Étapes pour les titres et pieds de page
BORDER, VSIZE, HSIZE, XPIXEL, YPIXEL, IBACK, CBACK, CTEXT, HTEXT, FTEXT	Options graphiques (GOPTIONS)

### 3.4. Exemples de sortie

Avec tous ces éléments, il est donc possible de générer un premier exemple de Dashboard KPI.

#### 3.4.1. Créer un premier Dashboard

Nous partons sur des valeurs imposées afin de produire un fichier HTML rapidement. Il s'agit ici de constater la facilité de mise en oeuvre et l'impact visuel de cette représentation. Le même Dashboard pourra être produit à partir d'une table et l'utilisation des macros-variables.

Ce sont six indicateurs qui seront représentés dans une page HTML créée avec l'ODS HTMLPANEL pour obtenir un tableau avec les indicateurs. Cet ODS est dédié à la création de tableaux regroupant des informations au format HTML.

##### 3.4.1.1. Détail du code

Nous utiliserons les éléments techniques suivants :

```
/* Les options graphiques : elles définissent le
moteur graphique et la taille en pixels de chaque
indicateur */
Goptions device=javaimg xpixels=200 ypixels=200 ;

/* Les options de la sortie HTMLPANEL : ces
options associées à cet ODS permettent de définir
le nombre de colonnes et la taille des bordures
par des macros-variables qui sont ensuite
exploitées dans le tagsets.
Le lecteur curieux pourra se référer à la
documentation en ligne */
%let panelcolumns = 3;
%let panelborder = 1;
%let embedded_titles = no;
```

```

/* Une sortie HTMLPANEL : les éléments
constituants de cet ODS permettent de spécifier
où enregistrer les fichiers.

On spécifie l'annulation des titres et pieds de
page graphiques */
ods tagsets.htmlpanel
  path = "c:\temp"
  body = "paneltest.html" (title="Principaux
indicateurs de suivi")
  nogtitle
  nogfootnote;

/*La définition du panneau de l'ODS : l'étape
Event permet de démarrer la création du tableau
*/
ods tagsets.htmlpanel event=panel(start);

/* Les indicateurs : au coeur sont placés les
indicateurs avec diverses options d'affichage */
proc gkpi mode=raised;
/*RAISED fournira un graphique en trois
dimensions */

speedometer actual=3000 bounds=(0 2000 4000
10000) / target=2000
/*Un accélérateur est utilisé pour les revenus
mensuels. En plus des seuils, une valeur cible
est affichée */

lfont=(f="Calibri" height=.5cm) /*La
police du libellé est définie */
format="eurox8.0" /*Les valeurs sont en
euros */
Label="Revenus du mois"; /*Un titre est
donné au KPI */
run;

proc gkpi mode=raised;
speedometer actual=-.05 bounds=(-.1 -.05 0 .05 .
1) / target = .05
format="percent5.1" /*Les valeurs sont en
pourcentage */
lfont=(f="Calibri" height=.5cm) label="Croissance
trimestrielle";
run;

proc gkpi mode=raised;
hslider actual=-6.7 bounds=(-10 -5 0 5
10) / /*Une barre à seuil est utilisée */
colors=(cxB2B2B2 cxB2B2B2 cxB2B2B2
cxB2B2B2) /*Le KPI sera en gris */
activecolors=(cxD06959 cxE1A05D cxBDCD5F
cx84AF5B) /*Le segment contenant la valeur
actuelle prendra une couleur spécifique */
lfont=(f="Calibri" height=.5cm) label="PBIT";
run;

proc gkpi mode=raised;
traffilight actual=598 bounds=(1500 900 600 0) /
/*Un feu tricolore est utilisé */
colors=(green yellow red) noavalue /*La
valeur actuelle n'est pas affichée */
lfont=(f="Calibri" height=.5cm) label="Conquête
client";
run;

proc gkpi mode=basic;
speedometer actual=12 bounds=(0 25 50
100) / /*Un accéléromètre est utilisé */
colors=(cx84AF5B cxF1DC63 cxD06959)
lfont=(f="Calibri" height=.5cm) label="Churn";
run;

```

```

proc gkpi mode=raised;
dial actual=.46 bounds=(0 .23 .46 .65 .79
1) / /*Une jauge est utilisée */

target=.9 nolowbound format="percent8.0"
/*Une valeur cible est affichée. Si la valeur
actuelle est sur un seuil,
le rond central prendra la couleur du segment
supérieur. Le format des données est en
pourcentage */

afont=(f="Albany AMT" height=.5cm) /*La
police de la valeur actuelle est définie */
bfont=(f="Albany AMT" height=.4cm) /*La
police des seuils est définie */
lfont=(f="Calibri" height=.5cm) label="Potentiel
de conquête globale";
run;

/*La fermeture de l'ODS : le tableau HTML est
fermé */
ods tagsets.htmlpanel event=panel(finish);
ods _all_ close;

```

### 3.4.1.2. Présentation du résultat

Le résultat est le suivant. L'ODS HTMLPANEL a permis de créer un tableau avec trois colonnes. C'est le nombre de procédures qui déterminera le nombre de cellules dans ces trois colonnes. Un titre est ajouté à la page HTML puis chaque procédure voit son résultat affiché dans une cellule.



1. Résultat de : `body = "paneltest.html" (title="Principaux indicateurs de suivi")`
2. L'aiguille présente la valeur actuelle
3. La flèche présente la valeur cible
4. La valeur actuelle est formatée en euros
5. La valeur actuelle formatée en pourcentage
6. La valeur actuelle est dans le premier segment. Celui-ci est donc colorisé
7. Les segments voient leur couleur inversée par rapport au visuel par défaut
8. La valeur actuelle est sur le seuil. Le rond central prend la couleur du segment supérieur

### 3.4.2. Exploiter des valeurs issues de tables SAS

Ces premiers développements vous ont intéressés ? Si vous souhaitez approfondir le sujet, je vous invite à prendre contact avec l'auteur pour obtenir l'article au complet.

### 3.5. Conclusion

En conclusion, la proc GKPI permet de créer des Dashboard très facilement afin de restituer une

information importante dans un espace restreint. Inutile dans ce cas d'opter pour une plate-forme BI si l'on souhaite diffuser cette page par ses propres moyens.

La procédure ne permet pas de lire une table donc il est nécessaire de définir les valeurs soit manuellement, soit avec un programme SAS qui renverra les valeurs souhaitées au travers de macros-variables.

#### 4. Développer son Dashboard avec SAS Enterprise Guide dans un environnement BI

L'objectif de cette deuxième partie est de construire un projet EG générant un Dashboard KPI. Tout comme la première partie, ce sont des KPI qui seront présentés dans une seule page pour obtenir un suivi d'activité synthétique.

La différence majeure porte sur le mode de diffusion des résultats. Il devient possible de diffuser largement le travail réalisé, grâce au référentiel de SAS BI.

SAS Enterprise Guide est l'outil dédié aux programmeurs dans un environnement SAS BI. En plus de programmes de qualification des données et d'analyse statistique il permet également de créer des applications stockées ou des rapports visibles depuis SAS Web Report Studio et de l'add-in Office. Grâce à ces deux fonctionnalités, nous illustrerons comment utiliser SAS Enterprise Guide pour mettre à disposition un Dashboard KPI depuis une procédure stockée dans le portail SAS et dans un rapport au format SRX pour SAS Web Report Studio et l'add-in SAS dans PowerPoint.

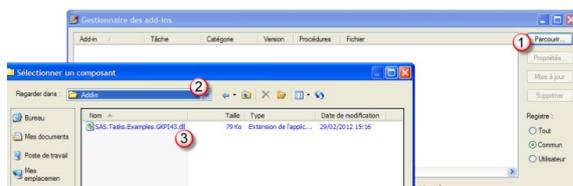
##### 4.1. Prérequis

La proc GKPI est accessible par un add-in qu'il convient de télécharger sur le site de SAS à l'adresse suivante : [Lien 88](#).

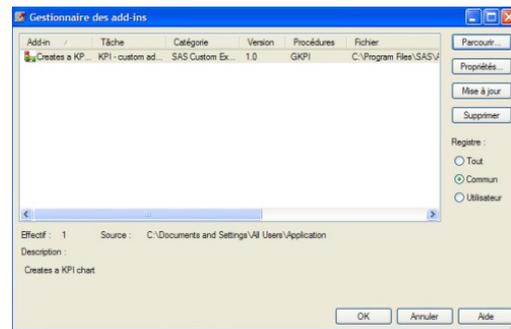
Il convient tout d'abord de télécharger la dll correspondant à sa version de SAS Enterprise Guide puis de renommer obligatoirement la dll après l'avoir déposée dans un répertoire accessible par l'outil.

Dll téléchargement	Dll exploitable par EG
fusion_36180_8_sas.dll	SAS.Tasks.Examples.GKPI.dll
fusion_36180_9_sas.dll	SAS.Tasks.Examples.GKPI43.dll

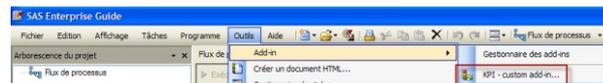
Depuis Enterprise Guide, il est alors nécessaire de déclarer cette dll depuis le menu Outils > Add-in.



L'assistant est alors correctement chargé :



L'assistant est accessible dans les add-in :

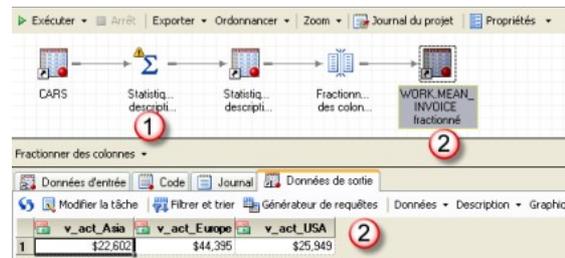


#### 4.2. Créer un projet KPI

Nous allons utiliser SAS Enterprise Guide pour concevoir un Dashboard KPI et le publier directement sur la plate-forme SAS. Nous allons voir que le rapport généré peut être statique (non actualisable) ou dynamique. Dans ce dernier cas, le rapport présentera les données rafraîchies à chaque ouverture du rapport sans avoir à le publier à nouveau.

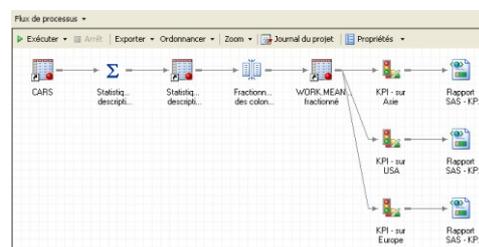
##### 4.2.1. Conception d'un projet KPI

Nous avons besoin d'une table de données contenant la valeur actuelle et la valeur cible d'un indicateur. Pour cela, calculons en premier lieu la moyenne des ventes par origine : à partir de la table CARS, constituons un petit projet qui calcule la moyenne des ventes par origine (1) des constructeurs puis nous transposons la table pour obtenir un indicateur par origine (2).

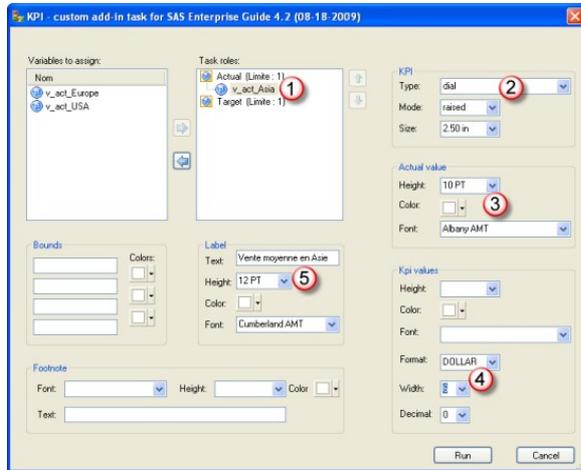


La valeur des ventes moyennes en Asie est donc de 22 602 \$.

À l'aide de l'assistant fraîchement téléchargé, nous générons trois KPI pour chacune de ces valeurs et obtenons trois rapports au format SRX. Pour rappel, le format SRX est un format de sortie SAS que les outils SAS peuvent ouvrir (du moins les applications Web et l'add-in).



L'illustration suivante permet de présenter l'interface de l'assistant de création de l'un de ces KPI. Cet assistant est basé en réalité sur la proc GKPI. Par conséquent, on retrouve les éléments vus au premier chapitre qui cette fois-ci sont mis à disposition avec des listes déroulantes et des boîtes de saisie.



1. La valeur actuelle prend la variable v\_act\_Asia
2. Un cadran est utilisé
3. La valeur actuelle présentée utilise une police Albany AMT de taille 10
4. Le format DOLLAR6.0 est appliqué aux valeurs
5. Un libellé mentionne l'origine

Le lecteur avisé remarquera immédiatement que les seuils ne sont pas renseignés. C'est un choix ici, mais le programmeur peut renseigner les valeurs sans toutefois les prendre dans une table comme il a été fait avec la valeur actuelle.

À noter que cet assistant possède une particularité intéressante lorsque les seuils sont laissés vides. Il analyse la valeur actuelle et impose des valeurs de seuils dans le code généré. Le code suivant illustre le travail fait dans l'assistant pour déterminer des seuils a priori. Cette partie n'est malheureusement pas modifiable.

```

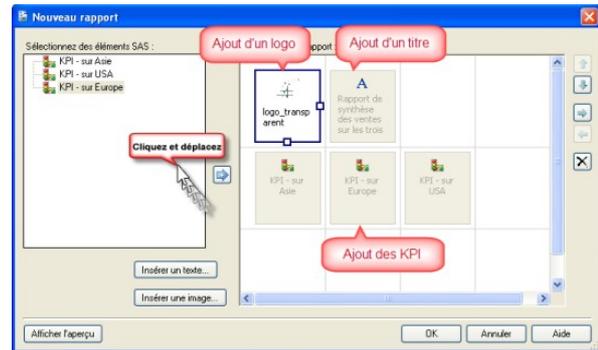
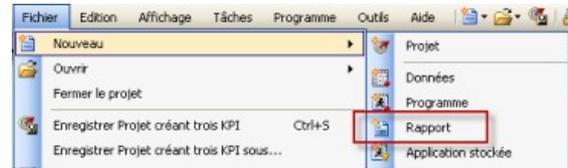
KPI - sur Asie
Données d'entrée Code Journal Résultats
Modifier la tâche Exporter Envoyer à Créer Propriétés

data actual_1;
set WORK.TMEAN_INVOICE;
call symput("mac_actual_1", 'v_act_Asia'n);
run;

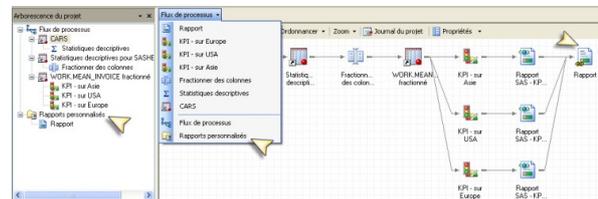
data bounds;
if &mac_actual_1 >=0 then do;
call symput("mac_b1",0);
if &mac_actual_1 <=1 then do;
call symput("mac_b2",.25);
call symput("mac_b3",.75);
call symput("mac_b4",1);

```

Dès lors que les sorties .srx sont créées, nous les regroupons dans un rapport. Cette fonctionnalité de SAS Enterprise Guide permet de créer des documents en regroupant des sorties.



Une fois que l'on clique sur OK, le rapport est généré et visible au niveau du flux de processus dans «Rapports personnalisés»

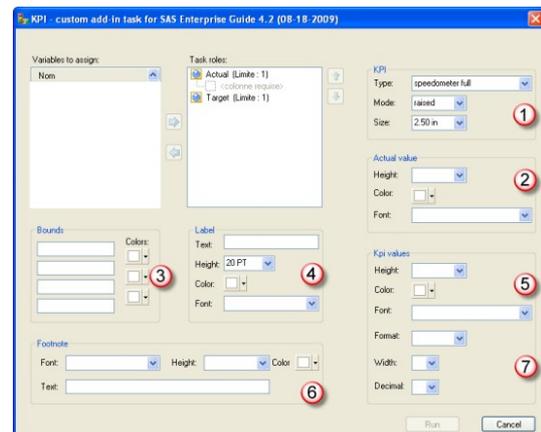


Nous visualisons le rapport en cliquant dessus:



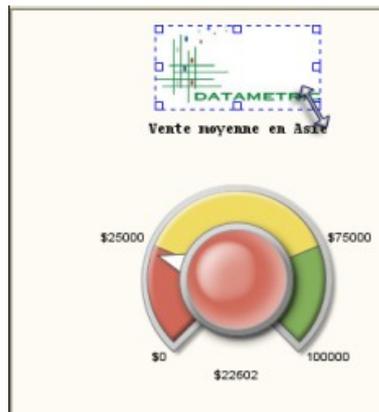
#### 4.2.2. Personnalisation de l'affichage

L'assistant permet de personnaliser les couleurs et les polices des valeurs présentées.



1. La valeur RAISED permet de créer un graphique en trois dimensions. La taille du KPI peut être modifiée également. Cela correspond à l'option `MODE=` et aux options `VSIZE` et `HSIZE`.
2. Police, couleur et taille de la valeur actuelle. Cela correspond à l'étape `AFONT=`.
3. La couleur par défaut des segments est modifiable. Cela correspond à l'étape `COLORS=`.
4. Police, couleur et taille du libellé du KPI. Cela correspond à l'étape `LFONT=`.
5. Police, couleur et taille des segments. Correspond aux étapes `BFONT=`.
6. Police, couleur et taille du libellé en pied de page. Cela correspond à une étape spécifique `FOOTNOTE`.
7. Format (type et longueurs) SAS donné aux valeurs. Correspond à une étape `FORMAT`.

Enfin, la mise en page de ce rapport peut être modifiée. Par exemple, ici nous redimensionnons le logo DATAMETRIC.



Le programmeur pourra obtenir d'autres informations en passant la souris sur chaque mot-clef de la procédure visible dans l'onglet Code.

Retrouvez la suite de l'article de DataMetric en ligne : [Lien 89](#)

```

KPI - sur Asie
Données d'entrée | Code | Journal | Résultats
Modifier la tâche | Exporter | Envoyer à | Créer | Propriétés
goptions reset=all device=javaimg vsize= 2.50 in hsize= 2.50 in;

proc gkpi mode=raised;
  dial actual=smac_actuel_1 bounds=(fmac_b1 fmac_b2 fmac_b3 fmac_b4) /
  label='Vente moyenne en Asie' lfont=(f= 'Cumberland AMT' h= 12 PT)
  colors=(cxD06959 cxF1DC63 cx84AF5B )
  bfont=(f= 'Times New Roman/B/I' h= 30 PT c= cxffff99)
  format= 'DOLLAR6.0';
run;
  
```

Keyword: FORMAT=

Contexte : option [DIAL instruction] FORMAT=

[Syntax: FORMAT="SAS-format"]

Specifies a SAS format for the boundary and actual values. The default format is BEST.

# Liens

- Lien 01 : <http://fr.wikipedia.org/wiki/Usenet>  
Lien 02 : <https://groups.google.com/forum/?fromgroups#%21forum/comp.lang.c++.moderated>  
Lien 03 : <https://groups.google.com/a/isocpp.org/forum/#%21forum/std-discussion>  
Lien 04 : <https://groups.google.com/a/isocpp.org/forum/#%21forum/std-proposals>  
Lien 05 : <http://cppnow.org/>  
Lien 06 : <http://www.developpez.net/forums/d1220785/c-cpp/cpp/communaute/nouveaux-forums-publics-comite-normalisation-cpp/>  
Lien 07 : <http://cpp.developpez.com/livres/?page=tous#L0201615622>  
Lien 08 : <http://cpp.developpez.com/livres/?page=tous#L020170434X>  
Lien 09 : <http://cpp.developpez.com/livres/?page=tous#L0201760428>  
Lien 10 : <http://cpp.developpez.com/>  
Lien 11 : <http://cpp.developpez.com/gotw/31>  
Lien 12 : <http://cpp.developpez.com/gotw>  
Lien 13 : <http://www.developpez.net/forums/d1229109/c-cpp/cpp/guru-of-the-week-francais/>  
Lien 14 : [http://cpp.developpez.com/faq/cpp/?page=destructeur#DESTRUCTEUR\\_virtuel](http://cpp.developpez.com/faq/cpp/?page=destructeur#DESTRUCTEUR_virtuel)  
Lien 15 : [http://cpp.developpez.com/faq/cpp/?page=constructeur#CONSTRUCTEUR\\_fonctions\\_virtuelles](http://cpp.developpez.com/faq/cpp/?page=constructeur#CONSTRUCTEUR_fonctions_virtuelles)  
Lien 16 : <http://cpp.developpez.com/redaction/data/pages/rubrique/cpp/gotw/030-039/gotw31/>  
Lien 17 : <http://blog.developpez.com/gpu/p10904/langages-a-frameworks/qt/qt5/opengl-dans-qt5/>  
Lien 18 : <http://doc.qt.nokia.com/4.7/graphicsview.html>  
Lien 19 : <http://qt.developpez.com/doc/4.7/vue-graphique-graphics-view/>  
Lien 20 : <http://qt.developpez.com/tutoriels/qt/qgraphicsscene/utilisation/>  
Lien 21 : <http://mac.developpez.com/livres/>  
Lien 22 : <http://access.developpez.com/faq/>  
Lien 23 : <http://www.developpez.net/forums/f45/logiciels/microsoft-office/access/>  
Lien 24 : <http://jdgayot.developpez.com/tutoriels/access/newsletter-cdo/fichiers/EmailingDvp.accdb>  
Lien 25 : <http://dico.developpez.com/HTML/144-Internet-spam.php>  
Lien 26 : <http://dico.developpez.com/HTML/1846-Generalites-blacklist.php>  
Lien 27 : <http://dico.developpez.com/HTML/1189-Internet-FAI-Fournisseur-dAcces-a-Internet.php>  
Lien 28 : <http://dico.developpez.com/html/3055-Infographie-IHM-Interface-Homme-Machine.php>  
Lien 29 : <http://jdgayot.developpez.com/tutoriels/access/newsletter-cdo/>  
Lien 30 : <http://bugs.jquery.com/ticket/9203>  
Lien 31 : <http://getfirebug.com/>  
Lien 32 : <http://wiki.commonjs.org/wiki/Promises/A>  
Lien 33 : <http://api.jquery.com/category/callbacks-object/>  
Lien 34 : <http://api.jquery.com/category/deferred-object/>  
Lien 35 : <http://fguillot.developpez.com/tutoriels/javascript/manipulation-callbacks-avec-jquery/>  
Lien 36 : [http://www.456bereastreet.com/archive/201105/styling\\_ordered\\_list\\_numbers/](http://www.456bereastreet.com/archive/201105/styling_ordered_list_numbers/)  
Lien 37 : <http://red-team-design.developpez.com/tutoriels/css/style-listes-ordonnees/fichiers/>  
Lien 38 : <http://www.yoyodesign.org/doc/w3c/css2/cover.html>  
Lien 39 : <http://www.red-team-design.com/before-after-pseudo-elements>  
Lien 40 : <http://code.google.com/p/chromium/issues/detail?id=54699>  
Lien 41 : <http://www.red-team-design.com/>  
Lien 42 : <http://www.red-team-design.com/css3-ordered-list-styles>  
Lien 43 : <http://red-team-design.developpez.com/tutoriels/css/style-listes-ordonnees/>  
Lien 44 : <http://www.inserthtml.com/2012/01/css3-multi-column/>  
Lien 45 : <http://www.inserthtml.com/>  
Lien 46 : <http://www.inserthtml.com/2012/01/css4-selectors/>  
Lien 47 : <http://inserthtml.developpez.com/tutoriels/css/specifications-css4/>  
Lien 48 : <http://wufoo.com/html5/>  
Lien 49 : <http://www.w3.org/TR/html5/urls.html#absolute-url>  
Lien 50 : <http://www.w3.org/TR/html5/common-microsyntaxes.html#valid-floating-point-number>  
Lien 51 : <http://www.w3.org/TR/html5/common-microsyntaxes.html#valid-simple-color>  
Lien 52 : <http://dev.w3.org/2006/webapi/XMLHttpRequest-2>  
Lien 53 : <http://dev.w3.org/2006/webapi/XMLHttpRequest-2/#the-formdata-interface>  
Lien 54 : <http://dev.w3.org/html5/spec/Overview.html#valid-e-mail-address>  
Lien 55 : <http://dev.w3.org/html5/spec/association-of-controls-and-forms.html#the-constraint-validation-api>  
Lien 56 : <http://dmouronval.developpez.com/tutoriels/html/formulaires-html5/fichiers/fichiers>  
Lien 57 : <http://dmouronval.developpez.com/tutoriels/html/formulaires-html5/>  
Lien 58 : <http://java.developpez.com/actu/43811/>  
Lien 59 : <http://www.oracle.com/technetwork/java/javase/overview/index-jsp-138218.html>  
Lien 60 : <http://www.oracle.com/technetwork/java/javafx/downloads/index.html>  
Lien 61 : <http://www.developpez.net/forums/d1218241/java/general-java/oracle-entame-migration-vers-java-7-grand-public/>  
Lien 62 : [http://fr.wikipedia.org/wiki/Théorie\\_des\\_graphes](http://fr.wikipedia.org/wiki/Théorie_des_graphes)  
Lien 63 : <http://www.diagram.ly/>  
Lien 64 : <http://www.jgraph.com/jgraphdownload.html>  
Lien 65 : <http://pbriand.developpez.com/tutoriels/java/JGraphX/Les%20bases/>  
Lien 66 : <http://www.eclipse.org/downloads/>  
Lien 67 : <http://download.eclipse.org/releases/indigo>  
Lien 68 : [http://jmini.developpez.com/eclipse\\_scout/articles/premiers\\_pas/](http://jmini.developpez.com/eclipse_scout/articles/premiers_pas/)  
Lien 69 : <http://www.developpez.com/actu/43505/>  
Lien 70 : <https://github.com/xamarin/XobotOS>  
Lien 71 : <http://www.developpez.net/forums/d1216259/java/general-java/java-mobiles/android/java-remplace-csharp-android/>  
Lien 72 : <http://cloud-computing.developpez.com/actu/43474/>  
Lien 73 : <http://www.youtube.com/watch?v=wKJ9KzGQ0w>  
Lien 74 : <http://www.developpez.com/actu/43664>  
Lien 75 : <http://www.developpez.com/actu/43145>  
Lien 76 : <http://www.developpez.net/forums/d1213650/logiciels/solutions-dentreprise/cloud-computing/google-drive-nouveau-service-stockage-google-ouvert/#post6642416>

- Lien 77 : <http://www.developpez.net/forums/d1213650/logiciels/solutions-dentreprise/cloud-computing/google-drive-nouveau-service-stockage-google-ouvert/#post6644063>
- Lien 78 : <https://drive.google.com/start#home>
- Lien 79 : <http://www.developpez.net/forums/d1213650/logiciels/solutions-dentreprise/cloud-computing/google-drive-nouveau-service-stockage-google-ouvert/>
- Lien 80 : <http://www.developpez.com/actu/43759>
- Lien 81 : <http://www.developpez.com/actu/44417>
- Lien 82 : <http://www.developpez.net/forums/d1225851/systemes/autres-systemes/mobiles/android-plus-dynamique-france-qui-os-dapres-at-internet/>
- Lien 83 : <http://community.pentaho.com/>
- Lien 84 : <http://sourceforge.net/projects/pentaho/files>
- Lien 85 : <http://sourceforge.net/projects/mondrian/files/>
- Lien 86 : <http://www.atolcd.com/actualites/detail-actualite/actualite/2/comparatif-etloopen-source-1.html>
- Lien 87 : <http://business-intelligence.developpez.com/tutoriels/presentation-pentaho/>
- Lien 88 : <http://support.sas.com/kb/36/180.html>
- Lien 89 : <http://datametric.developpez.com/tutoriels/sas/sas-foundation-sas-bi-creer-dashboard-kpi/>