doc.ubuntu-fr.org

imagemagick - Documentation Ubuntu Francophone

ImageMagick est un logiciel **en ligne de commande** très puissant de manipulation d'images dans pratiquement tous les formats existants. Il consiste en une suite d'outils permettant par exemple de changer le format, l'échelle, l'orientation, rajouter une bordure ou du texte, appliquer un filtre, fusionner plusieurs image, animer une suite d'images, etc. Il est possible d'utiliser ces différents outils dans des programmes écrits en C, C++, Ruby, Python, Perl, etc.

Installation

Il vous suffit, pour cela, d'installer le paquet imagemagick.

Les différents outils

Les outils formant ImageMagick possèdent de nombreuses options étendant très loin leurs possibilités. La plupart de ces options sont communes à plusieurs outils. Ainsi, l'option *-resize 50%* permet :

- avec **display**, de visualiser une image à la moitié de ses dimensions ;
- avec **convert**, de créer une image de dimensions moitié moindre.

Pour afficher vos tests, si vous êtes sous gnome , vous pouvez utiliser Eye of GNOME :

```
eog monimage.png
```

Mais, quel que soit votre environnement de bureau, **display** est plus léger et plus puissant.

Voici ci-après la liste des différents outils ainsi que quelques exemples simples, le cas échéant.

Display

Display affiche une image à l'écran :

```
display image.png
```

En cliquant sur l'image apparue à l'écran, vous aurez accès à une interface graphique sommaire qui vous permettra tout de même de nombreuses modifications de l'image ou de son affichage et même un diaporama du dossier. Utilisez la touche « q » pour quitter l'application.

Convert

Convert permet la modification d'une ou plusieurs images, par exemple :

- créer une image TGA à partir d'une image JPEG :

  ```
  convert image.jpg image.tga
  ```

- convertir une capture PNG en JPEG **et** la compresser à 20 %. Cela est **fort utile** pour ne pas alourdir inutilement vos demandes d'aide sur le forum :

  ```
  convert -quality 20 image.png image.jpg
  ```

- créer un fichier GIF animé à partir de plusieurs images PNG :

  ```
  convert images_*.png anime.gif
  ```

Cette commande permet la création d'un GIF animé d'assez bonne qualité à partir de photos JPEG (l'option `-delay` permet de spécifier un temps en centièmes de secondes, l'option `-loop 2` permet de faire 2 boucles des photos et 0 une lecture en continue) :

```
convert -delay 50 -loop 0 *.jpg animation.gif
```

- redimensionner en forçant un changement de proportions :

  ```
  convert test1.jpg -resize 500x1050\! test1b.jpg
  ```

- créer un document PDF à partir d'une série d'images png :

```
convert -compress jpeg images_*.png document.pdf
```

L'option `-compress jpeg` force la compression des images au format JPEG plutôt qu'au format MTIF par défaut. cela pour effet d'offrir un fichier PDF nettement plus petit et plus compatible avec Adobe Reader.

- diminuer les dimensions d'une image tout en effectuant une rotation de 90° dans le sens horaire :

```
convert images.jpg -resize 50% -rotate 90 image.jpg
```

- tourner les images en fonction de leur orientation donnée dans le format EXIF, comme par exemple les images issues d'un appareil photo numérique. L'image n'est retournée que si elle a été prise en portrait, si elle a été prise en paysage, celle-ci n'est pas modifiée :

```
convert image.jpg -auto-orient image_retournee.jpg
```

- ajouter un *copyright* (« © Ubuntu » par ex.) ou autre *watermark* sur les photos : des scripts ont déjà été faits, facilitant l'utilisation des lignes de commandes, en particulier pour l'ajout d'un *copyright* image transparente : voir dans ce post de forum avec ce script de Fake ;

- découper une image selon le principe de l'emporte-pièce. Les paramètres hauteur et largeur permettent de fixer la hauteur et la largeur de l'image finale (dans la limite des dimensions de l'image originale), les paramètres x et y permettent de localiser le coin supérieur gauche de l'image finale par rapport au coin supérieur gauche de l'image initiale. Pour découper une image de 100 px de hauteur et de 50 px de largeur située à 10 px du bord gauche et à 20 px du bord supérieur de « initial.png » :

```
convert -crop 50x100+10+20  initial.jpg final.jpg
```

- découper une image de dimensions 55 × 110 en 18 morceaux, 10 morceaux de 20 × 20, 2 morceaux de 20 × 10, 5 morceaux de 15 × 20 et 1 morceau de 15 × 10. En effet, si x et y sont omis, l'image est intégralement découpée en morceaux de dimensions hauteur×largeur, la découpe se faisant de gauche à droite et de haut en bas, à partir du coin supérieur gauche (il est possible de remplacer hauteur×largeur et x et y par un % de la largeur. x et

y peuvent être négatifs, dans ce cas le point de départ de la découpe se fera en dehors de l'image d'origine, seule la partie correspondant à des pixels de l'image d'origine étant restituées). Les parties les plus à droite et les plus en bas peuvent être de dimensions inférieures à la taille de la découpe pour s'adapter à la dimension de l'image initiale :

```
convert -crop 20x20 initial.jpg final.jpg
```

- transformer un dossier d'images :

```
convert *.BMP -set filename:f '%t.png' +adjoin '%[filename:f]'
```

Mogrify

Mogrify est utilisé pour apporter la même modification à plusieurs images. par exemple, pour augmenter le contraste d'une série de photos :

```
mogrify -sigmoidal-contrast 5,50% webcam-shot*.png
```

Il est également possible d'utiliser mogrify pour redimensionner une image (ou un lot d'images), avec la commande suivante :

```
mogrify -resize 800x600 image.jpg
```

ou pour toutes les images d'un dossier :

```
mogrify -resize 800x600 *.jpg
```

Attention, *mogrify* réécrit sur les images d'origine, pensez à faire des tests avant de lancer la commande finale ou utiliser l'option `-path` pour que le résultat de la ligne de commande s'écrive dans un **autre** répertoire :

```
mogrify -resize 800x600 -path /autre_repertoire *.jpg
```

Ou encore, conversion et changement de proportions :

```
mogrify *.png -resize 500x1050\! -path /autre_repertoire  *.jpg
```

Ce dernier traitement par lot est particulièrement utile pour corriger en un clin d'oeil des captures d'écran si, par exemple, la numérisation de vos cassettes VHS n'a pas respecté les proportions originelles.

Identify

Identify donne des informations sur l'image.

- Pour des informations sommaires :

```
identify image.jpg
```

- Pour des informations complètes :

```
identify -verbose image.jpg
```

- Pour les images contenant un grand nombre de couleurs, la commande précédente pourra renvoyer des centaines de lignes d'informations. Pensez à l'associer à la commande **less** ou **grep** à travers un pipe pour plus de lisibilité. Ainsi, pour connaître le taux de compression de votre JPEG :

```
identify -verbose image.jpg | grep Quality
```

- Pour spécifier un format d'affichage (idéal pour le scripting)

```
identify -format "largeur de %w px" image.jpg
largeur de 51 px
```

Import (faire des captures d'écran)

Import dispose d'un très grand nombre d'options, pour plus d'informations reportez vous à la page de documentation du projet.

Capture d'écran interactive

Utilisez la commande :

```
import image.png
```

Le curseur de la souris se transformera alors en « croix ». Vous pourrez alors :

- cliquer une fois dans l'écran pour capturer tout l'écran ;

- tracer un cadre avec un « cliquer-glisser » qui capturera la zone définie.

Principales options

-window

Cette option vous permet de capturer l'écran entier :

```
import –window root image.png
```

-pause

Cette option vous permet d'attendre le nombre spécifié de secondes avant que le curseur ne change de forme.

```
import –pause 10 image.png
```

-crop

Utilisable en conjonction avec `–window` , cette option vous permet de spécifier exactement quelle zone de l'écran doit être capturée.

Sélectionner une zone de 800×600 pixels en partant du coin supérieur gauche de l'écran :

```
import –window root –crop 800x600 image.png
```

Sélectionner une zone de 800×600 pixels en partant du point situé 150 pixels à plus à droite et 100 pixels plus bas que le coin supérieur gauche de l'écran :

```
import -window root -crop 800x600+150+100 image.png
```

Le manuel pour plus d'information.

Animate

Animate permet la visualisation d'animations.

- Pour visionner un GIF animé :

  ```
  animate images.gif
  ```

- Pour animer une série de photos à raison d'une par seconde :

  ```
  animate -delay 100 *.png
  ```

Compare

Compare crée, à partir de 2 images, une troisième qui représente la différence entre les 2 premières. Utile pour savoir où ont été opérées des modifications :

```
compare imageA.png imageB.png difference.png
```

Composite

Pour faire se chevaucher ou mélanger des images.

Il semblerait que la superposition puisse se faire avec tout une palette d'effets (transparence, etc ...) qui restent à expliciter. Composite permet, par exemple, d'ajouter une signature qui peut être elle même une image (si elle est transparente, la superposition le sera également). Il est également possible d'écrire directement sur une image *via* "annotate" mais avec une image personnalisée c'est plus joli.

- `-compose` : Permet de définir la composition de l'image finale, c'est à dire position des deux images l'une par rapport à l'autre et le mode de superposition ;

- `-geometry` : Permet de déplacer l'image du dessus d'un certain nombre de pixels ("offset") sur l'axe des abscisses et l'axe des ordonnées, les valeurs positives emmènent vers le centre de l'image, les valeurs négatives s'en éloignent ;

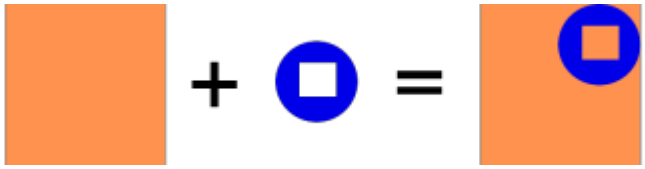
- `-gravity` : le placement de l'image superposée ou le point de départ du calcul de `-geometry` . Valeurs possibles :

  - coin supérieur gauche (valeur par défaut) : `Northwest`
  - coin supérieur droit : `NorthEast`
  - coin inférieur gauche : `SouthWest`
  - coin inférieur droit : `SouthEast`
  - milieu du bord supérieur : `North`
  - milieu du coté droit : `East`
  - milieu du bord inférieur : `South`
  - milieu du bord gauche : `West`

- centre : `Center`

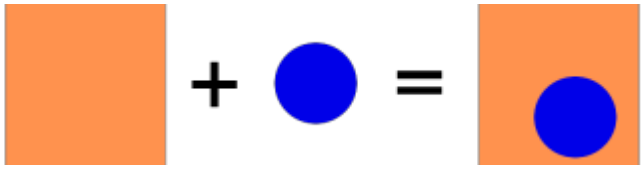
Exemple : ajout d'un motif en haut à droite d'une image :

```
composite -compose Over -gravity NorthEast cercle_bleu.png
carre_orange.jpeg Image_résultante_finale.jpeg
```



Ajout d'un motif centré, décalé de 40 pixels vers le bas et de 20 pixels vers la gauche :

```
composite -compose Over -geometry +20+40 -gravity Center cercle_bleu.png
carre_orange.jpeg Image_résultante_finale.jpeg
```



Superposition simple de deux motifs comportant des zones transparentes (placement en haut à droite) :

```
composite -compose Over -gravity NorthEast cercle_bleu_perce.png
triangle_rouge_perce.png  Image_résultante_finale.jpeg
```

Superposition gérant la transparence de deux motifs comportant des zones transparentes (placement en haut à droite) :

```
composite -compose Atop -gravity NorthEast cercle_bleu_perce.png
```

```
triangle_rouge_perce.png  Image_résultante_finale.jpeg
```

Options de composition (liste détaillée)

Dans la suite de la description des commandes de Composite image supérieure, première image, image "du dessus" sont synonymes et correspondent à l'anglais "overlay" ; image inférieure, deuxième image et image "du dessous" sont synonymes et correspondent à l'anglais "background".

Les options possibles :

-compose

Permet de définir la composition de l'image, c'est à dire la manière dont la superposition s'effectue. Plusieurs méthodes sont possibles (schémas visibles ici) :

Méthodes de superposition :

1. Over : superposition la plus simple, l'image "du dessus" (qui vient en premier dans la ligne de script) se superpose sur celle du dessous, la taille de l'image finale est celle de l'image du dessous. Dans l'image finale, on aperçoit l'image du dessous à travers les parties transparentes de l'image supérieure ;

2. Dst_over : la première image s'affiche sous la seconde (dans l'ordre du script) mais la taille de l'image résultante est celle de la seconde image (c'est à dire l'image inférieure) ;

3. Src : L'image "du dessous" est recouverte par l'image "du dessus" MAIS toutes les meta-données de l'image résultante seront celles de l'image inférieure et la taille de l'image finale sera celle de l'image inférieure. Si l'image de recouvrement (supérieure) est de taille inférieure à l'image "du dessous", le reste de l'image "du dessous" sera soit transparent (si l'image "du dessous" dispose d'un canal alpha), soit noir ;

4. Copy : Même résultat que Src sauf pour la partie de l'image de fond ("du dessous") qui "dépasserait" autour de l'image supérieure : l'image de fond apparait sous l'image du dessus si elle est de taille supérieure. L'effet est équivalent à celui de "Over" sauf si l'image supérieure dispose d'une zone transparente : l'image finale sera transparente à cet endroit (alors qu'avec "Over" l'image inférieure apparaîtrait dans la zone transparente) ;

5. Dst_in : l'image supérieure est utilisée comme masque pour l'image inférieure : seule la partie de l'image inférieure qui serait recouverte par l'image du dessus apparaît dans l'image finale, mais c'est bien le motif de l'image inférieure qui est affiché. Le reste de la surface de l'image finale est transparente. L'image supérieure est utilisée comme un emporte-pièce. L'image supérieure ne doit pas être une image en nuances de gris car c'est son canal alpha qui est utilisé par imagemagick ;

6. Dst Out : cette commande est complémentaire de la précédente. L'image finale est analogue à l'image inférieure, le motif de l'image supérieure apparaissant comme une zone transparente. En reprenant l'image d'un emporte-pièce Dst_in représente la partie enlevée et Dst_out la partie "qui reste", la partie enlevée laissant un "trou" transparent. En superposant les deux images (elles ont toutes deux les dimensions exactes de l'image inférieure d'origine), on retrouve l'image inférieure d'origine.

7. ATop : cette commande est équivalente à "Over" si l'image de fond est totalement opaque. Si l'image de fond ("du dessous") comporte des zones transparentes, celles-ci demeurent transparentes dans l'image finale, seules les parties opaques situées sous l'image supérieure "du dessus" sont recouvertes ;

8. Dst ATop : cette commande va superposer sur la première image les parties non transparentes de la seconde, l'image finale ;

9. Xor : rend transparente les zones colorées des deux images superposées. L'image finale comportera les zones de transparence communes aux deux images (l'une sous l'autre), les zones opaques des deux images qui correspondent à une zone transparente sur l'autre image et des zones transparente dans les zones qui sont opaques dans les deux images ;

10. Clear : cette commande se contente d'effacer les couleurs de l'image "du dessous" et ne prend pas en compte l'image supérieure. C'est un outil pour la création de canevas transparents ou noirs dans les scripts complexes. Si le canal alpha est clôt, l'image créée sera noire. L'option spéciale "compose:outside-overlay" limite l'effacement à la surface de l'image "du dessus" (équivalent de "Dst_out" pour une image supérieure ne comportant aucune zone de transparence ;

11. Dst : cette commande ne fait rien du tout. Elle est utile dans les scripts complexes pour annuler facilement une composition transparente.

Méthodes mathématiques :

Opérations sur les couleurs (multiplications, additions, etc.). Généralement réservé aux images en nuance de gris (mais possible pour la couleur quand même).

1. Multiply - multiplication (première x deuxième) : multiplication des valeurs de chaque pixel (dont la valeur varie entre 0 - noir et 1 - blanc), si une image est en blanc pur, le résultat sera l'autre image. Exemple : dégradé du blanc au noir horizontal x dégradé vertical : dégradé sur un angle ;

2. Screen : équivalent de Multiply mais les images sont passées en négatif avant traitement puis le résultat est lui-même passé en négatif. Si l'une des deux images est en noir pur, le résultat sera équivalent à l'autre image ;

3. Bumpmap (multiplication des nuances de gris) : l'opération va assombrir la seconde image dans toutes les zones où la première est sombre ;

4. Divide - division: l'opération va éclaircir la seconde image dans toutes les zones où la première est sombre ;

5. Plus - addition : addition des couleurs, les couleurs de mélangent. Des résultats inattendus peuvent se produire si la somme mathématique de la valeur des couleurs est supérieure à la valeur maximale admise. La couleur résultante sera donc blanche (valeur 1). L'opérateur additionne aussi la valeur du canal alpha (c'est le seul à le faire) ;

6. ModulusAdd : équivalent à Plus sauf en cas de dépassement de la valeur maximale admise, les pixels seront noirs et non blancs ;

7. Minus : soustraction d'une image à l'autre. Les valeurs négatives seront noires (0) ;

8. ModulusSubtract : équivalent à Minus sauf pour les dépassements : soustraire du blanc à du gris donnera du gris ; des résultats imprévus sont également possibles dans la soustraction des canaux alpha ;

9. Difference : l'image résultante correspond à la différence absolue des valeurs des couleurs. La soustraction de blanc au noir produira du blanc alors que la soustraction de couleurs identiques donnera du noir. Cette commande permet de comparer deux images et de calculer un ratio de différences ;

10. Exclusion (Soustraction d'images sauf les gris - la formule de calcul est la suivante : Dessus + Dessous - 2*Dessus*Dessous). Blanc - blanc donnera du noir, noir - noir aussi, gris - gris du gris, blanc - noir donnera du blanc. Les zones seront d'autant plus claires que le gradient entre les deux images sera élevé ;

11. Lighten (sélection des couleurs plus claires) : Les zones les plus claires des deux images sont sélectionnées (si première > seconde alors première sinon seconde)

12. Darken : inverse de Lighten.

Méthodes agissant sur la luminosité :

1. Overlay (ajout de couleurs à une image en nuances de gris) : Méthode composite d'assombrissement (Multiply) et d'éclaircissement (ligthten). Préserve le blanc et le noir purs de l'image de fond et tinte les gris à partir des couleurs de l'image "du dessus" (Formule : If dessous ⇐ 0.5 then (2*dessus*dessous) else (1-2*(1-dessus)*(1-dessous)) ;

2. Hard_Light : équivalent de Overlay avec inversion des première et seconde image ;

3. Soft Light : le traitement est équivalent à Hard Light mais les gradients sont réduits, de fait le résultat est plus proche de Overlay ;

4. Pegtop_Light : très proche de Soft Light, la formule est encore adoucie et le traitement plus aisé ;

5. Pin_Light : destiné à mieux préserver les demi-teintes, les valeurs les plus sombres et les plus claires sont restreintes. Le résultat est moins lissé ;

6. Linear_Light : Méthode de fusion simple qui produit davantage de blancs et de noirs purs. La formule est : (2*Première + Deuxième) - 1 ;

7. Vivid_Light (même méthode que Photoshop 7). Refinement mineur de Linear_light.

Limite les valeurs extrêmes. Formule :If Première ⇐ 0.5 then 1-Deuxième)/(2*Première) else Deuxième/(2*(1-Première)) ;

8. Linear_Dodge (Photoshop 'Add' Compose). Proche de Plus pour les images opaques, différent pour les images semi-transparentes. Formule : Première + Deuxième ;

9. Linear_Burn (Photoshop 'Subtract' method). Equivalent au passage en négatif des deux images avant Lienar-Dodge puis à nouveau passage en négatif de l'image résultante. Formule : Première + Deuxième - 1 ;

10. Color_Dodge (protection de l'exposition à la lumière). L'image "du dessus" va "protéger" l'image "du dessous" de la lumière. La superposition d'une image en blanc pur va "blanchir" toute l'image du dessous, une imange en noir pur sera sans effet. Formule : Deuxième / (1-Première) ;

11. Color_Burn : effet inverse de Color_dodge. Les couleurs de l'image du dessous situées sous les parties claires de l'image du dessus seront protégées, les parties situées sous les zones sombres seront assombries. Formule : 1 - ((1-Deuxième) / Première).

Méthodes de copie de canaux :

Ces méthodes assurent le transfert des information d'un cannal d'une image vers celui d'une autre image ;

1. Copy_Opacity (fixer la transparence à partir du masque en nuances de gris) : Quand la première image n'a pas de canal alpha, l'opérateur va remplacer le canal alpha de la seconde image avec le canal des nuances de gris de la première. Tout ce qui est noir dans la première image sera transparent, tout ce qui est blanc opaque ;

2. Copy_Red, Copy_Green, Copy_Blue : Copie du canal de couleur sélectionné de la première image dans la seconde - Équivalent aux méthodes précédentes ;

3. Copy_Black : Copie du canal noir quand il existe (images CMYK), aucun effet sinon ;

4. Hue (copie de la nuance d'une image RGB) : Réservé aux images RGB à espace de couleur. Les valeurs de saturation S et de luminance L (ou Y?) de la seconde image ne sont pas modifiées ;

5. Saturate : Copie de la Saturation S sans modification des valeurs des nuances de couleur hue H et de la luminance L (ou Y?);

6. Luminize copie de la luminance d'une image RGB sans modification des valeurs des nuances de couleur hue H et de la saturation S.

7. Colorize copie de la nuance d'une image RGB et de la de la Saturation S, la luminance L (ou Y?) de la seconde image n'est pas modifiée.

-dissolve

Dissolution d'une image dans l'autre. Cette méthode effectue une superposition contrôlée de la première image sur la seconde, en ajustant la transparence de la première sur la seconde. Il est également possible d'organiser la dissolution progressive de la seconde image (valeurs de transparence de 100 % à 200 %). Syntaxe :

```
composite -dissolve pourcentage Première Deuxième Image_résultante
composite -dissolve Première_pourcentage}x{Deuxième_pourcentage}
Première  Deuxième  Image_résultante
convert Deuxième  Première   -compose dissolve -define
compose:args=Première_pourcentage,Deuxième_pourcentage -composite
```

```
Image_résultante
```

-blend : fusion d'images

Dissolution des deux images, les deux images sont traitées de la même manière (pas de superposition, plutôt mélange), en fonction du pourcentage fixé pour chacune dans la commande. Syntaxe :

```
 composite -blend pourcentage Première  Deuxième  image_résultante
 composite -blend première_pourcentagexdeuxième_pourcentage Première
Deuxième  image_resultante
 convert Deuxième  Première   -compose blend -define
compose:args=première_pourcentage,deuxième_pourcentage -composite
image_resultante
```

-watermark ("modulate" compose method)

Dégradation d'image et l'aposition d'un filigrane pour protéger le copyright. Syntaxe :

```
composite -watermark valeur_luminosité[xvaleur_saturation] Première
Deuxième  image_resultante
```

La première image est une image noir et blanc avec canal alpha masqué qui est utilisée pour éclaircir ou assombrir la deuxième image selon un pourcentage (0 : aucun effet, 100 : superposition complète) fixé dans la commande.

-composite

Utilisation d'un masque pour limiter la superposition d'une image sur une autre. Le masque de composition permet d'utiliser une troisième image qui va limiter la zone affectée par une superposition de la première image sur la secone. La taille de l'image finale résultante sera celle de la seconde image. Les zones transparentes du masque ne seront pas prises en compte, les zones noires "protègeront" la seconde image, les zones blanches seront affectées par l'opération de superposition effectuée avec la première image.

-tile

La première image est reproduite comme un motif en plusieurs exemplaires de manière à couvrir la

deuxième image. Syntaxe :

```
composite  -tile star.gif   netscape:   tile.gif
```

Méthodes spéciales de composition

= Mathematics =

Utilisation de 4 valeurs numériques pour appliquer des traitements numériques à la carte. (Formule : A*Première*Deuxième + B*Première + C*Deuxième + D) Syntaxe :

```
 '-compose Mathematics -set option:compose:args -1,1,1,0 -composite
image_résultante.png
```

Certaines combinaisons permettent d'obtenir l'équivalent de Multiply, LinearLigth, etc.) Méthode Compose Arguments de Mathematics

```
  Multiply              1,0,0,0
  Screen                -1,1,1,0
  Exclusion             0,1,1,-1
  Linear_Dodge          0,1,1,0
  Linear_Burn           0,1,1,-1
  Linear_Light          0,2,1,-1
```

= Change_Mask =

Rend transparents certains pixels de l'image résultante finale en fonction de la valeur du Fuzz Factor. Utile pour reconstruire la transparence d'une image qui a recouvert un arrière-plan complexe suffisamment différent pour que la fonction puisse agir.

```
convert Deuxième_superposé.gif Deuxième.gif -compose ChangeMask
-composite  Deuxième_removed.png
```

Les images JPEG couleur ont souvent de légèress variations de couleur résultant des pertes liées à la compression, le réglage du Fuzz Factor doit être faible pour cibler les couleurs proches.Il est parfois utile d'inverser les deux images pour reconstruire la transparence à partir du "trou" de l'arrière plan plutôt que d'agir sur l'image superposée.

Conjure

Interprète et exécute un script écrit en Magick Scripting Language (MSL).

Montage

Pour faire une composition de plusieurs images.

L'option '-geometry' donne la taille de chaque image en pixels qu'il faudra introduire. S'utilise comme suit : -geometry "largeur"x"hauteur". L'option '-tile' donne la disposition des images sur la grande unifiée : -tile "colonnes"x"lignes". En pratique :

```
montage -geometry "largeur"x"hauteur" -tile 2x2 *.jpg together.jpg
```

donnera autant d'images qu'il le faut pour assembler tous les jpeg du dossier courant à raison de quatre par page dans des jpeg dont le nom commence par together (together-0.jpg, together-1.jpg, etc.).

Très utile pour imprimer des photos en format carte sans gaspiller de papier... mais il faut découper après (utilisation d'un massicot recommandée).

Stream

Pour pouvoir manipuler de grandes images.

FAQ

Où trouver imagemagick-devel ?

Le paquet imagemagick pour ubuntu ne contient pas tous les outils de ImageMagick, et notamment la librairie « devel » qui est requise par certains autres logiciels s'appuyant sur imagemagick, comme par exemple la gemme RMagick de Ruby, ou encore l'extension de PHP5 qui s'appuie sur imagemagick. Les librairies à installer sont en fait libmagickwand-dev et libmagickcore-dev.

J'obtiens une erreur 'Échec de la délégation' lorsque je lance une commande

Lorsque vous lancez une commande, ImageMagick peut délèguer des opérations à d'autres programmes. Il est donc nécessaire que ces derniers soient installés.

Pour obtenir la liste de ces délégations, tapez : `convert -list delegate`

Pour obtenir la liste des délégations pour un format particulier, par exemple le svg, tapez :

```
convert -list delegate | grep 'svg = '
```

ImageMagick vous donnera alors les programmes qu'il utilise. Pour le format svg, cette commande peut retourner : `svg ⇒ "rsvg-convert" -o "%o" "%i"`

En tapant "rsvg-convert", votre shell peut alors vous indiquer le paquet à installer pour lancer cette commande :

```
rsvg-convert
Le programme « rsvg-convert » n'est pas encore installé. Vous pouvez
l'installer en tapant :
sudo apt-get install librsvg2-bin
```

Donc librsvg2-bin pour le support des images svg.

Voir aussi