

Le lecteur Freepaper 2

visualiser les fichiers PDF dans les pages WEB

Freepaper 2

A blue, stylized logo for version 0.9.0, featuring the numbers '0.9.0' in white with a blue outline, set against a blue background with a white, textured, cloud-like pattern.

1 Pourquoi FreepaperR

Pagegangster, Motion Paper, scribd.com, myclick.com..... tous ces sites proposent la publication en ligne de vos documents PDF.

Il faut pour cela au préalable uploader ses documents sur les serveurs du prestataire choisi, un peu à la manière des YouTube ou Flickr.

Cela ne pose pas de problème dans la majorité des cas. Cependant, on ne souhaite parfois pas que ses documents deviennent publics, qu'ils soient analysés par des robots ou encore que de la publicité y soit insérée.

FreepaperR aussi permet la visualisation en ligne de fichiers PDF, mais il s'installe sur son propre serveur et les documents que l'on affiche ne quittent jamais le domaine, ne sont jamais altérés, sont toujours disponibles...

Le principe est le suivant:

- Le document à visualiser est déjà situé sur le serveur
- Il est converti grâce à la boîte à outil (GPL) [swftools](#). Ainsi, on obtient un nouveau fichier, qui est la version SWF du fichier PDF original
- On utilise un FreepaperR pour naviguer dans le fichier généré

Remarque : Cette méthode de présentation d'information ne permet pas, à la différence de l'écriture textuelle le référencement par les moteurs de recherche, le document étant « caché » au monde par le lecteur SWF. Il y a toujours moyen de trouver des artifices tels qu'insérer dans la page du contenu textuel caché, mais cela est néanmoins à déconseiller.

Pour faire fonctionner cet outil, il faut un serveur WEB (php 4 ou 5 sont conseillés pour bénéficier de l'intégralité des fonctionnalités de FreepaperR).

Bien entendu, il faut idéalement pouvoir exécuter des routines de la boîte à outils swftools sur le serveur, ce qui est sans doute le point le plus délicat. Mais nous verrons par la suite des méthodes alternatives permettant de contourner ce point, notamment dans le cas des hébergements mutualisés sans accès SSH.

2 Implantation du lecteur FreepaperR

Extraire le contenu de l'archive freepaper2.0.9.0.zip, dans le dossier de votre choix. Si vous souhaitez une structure de fichier différente de celle proposée, il faudra penser à éditer les deux variables situées en début du fichier freepaper2-min.js :

```
var m_freepaper2_swfUrl="swf/freepaper2.0.9.0.swf"; // URL du fichier "freepaper2.0.9.0.swf"
var m_freepaper2_phpURL="php/freepaper.php"; // URL du script PHP freepaper.php
```

On utilise la boîte à outils javascript **swfobject 2.2**.

Note : la documentation sur l'usage de swfobject ainsi que le téléchargement de l'archive se fait à l'adresse suivante : <http://code.google.com/p/swfobject/>

1 Importer dans la partie <head> de la page les fichiers javascript **swfobject.js** et **freepaper2-**

min.js

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr" dir="ltr">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Freepaper 2 version 0.9.0</title>
  <script type="text/javascript" src="javascript/swfobject.js"></script>
  <script type="text/javascript" src="javascript/freepaper2-min.js"></script>
  <link rel="stylesheet" href="css/freepaper2.css" type="text/css" media="all">
</head>
```

2 Ajouter le code indiquant le point d'insertion du lecteur FreepaperR dans la page :

```
<div id="freepaper1">
  <a href="http://www.adobe.com/go/getflashplayer">
    
  </a>
</div>
```

Si le plugin Flash est manquant ou si la version est trop ancienne, alors cette partie du code sera utilisée pour indiquer un lien vers le plugin adéquat.

L'élément important est **id**, dans notre exemple "**freepaper1**" qui permettra d'indiquer à la boîte à outils swfobject où implanter, si possible, le lecteur FreepaperR. Ainsi, le conteneur <div> d'id freepaper1 devrait être remplacé par un objet SWF.

3 Créer le lecteur **FreepaperR**, avec ses options

```
<script type="text/javascript">
//Options pour l'insertion d'un lecteur FreepaperR
var flashvars = {
    docURL : "documents/gazetteCDM_2_200810.pdf" //obligatoire
};
var params = {
    width:600, //optionnel
    height:800, //optionnel
    scale: "noScale", //optionnel
    wmode : "opaque", //optionnel
    allowFullScreen:"true" //obligatoire
};
var attributes = {
    altContentId:"freepaper1", //optionnel
    trace:true //optionnel
};
//Insertion d'un lecteur FreepaperR
freepaper2.embedDoc(flashvars,params,attributes);
</script>
```

docURL : obligatoire - chemin du document à afficher.

Si le document est un document **swf**, alors il est directement affiché. On n'utilise alors pas de script php (php n'est donc pas nécessaire dans ce cas),

Si le document a une extension **pdf**, ce fichier est analysé par un script php et éventuellement converti (php nécessaire),

width : optionnel - la largeur de l'écran FreepaperR (valeur par défaut, 600)

height : optionnel - la hauteur de l'écran FreepaperR (valeur par défaut, 800)

allowFullScreen:"true" : obligatoire : permet le mode plein écran

scale: "noScale" : optionnel – l'objet FreepaperR conserve son échelle en cas de redimensionnement

wmode : "opaque" : facultatif : Une valeur « opaque » ou « transparent » permet de placer le lecteur dans le système de couche du DOM (ce qui l'autorise à être affiché en dessous d'autres élément HTML).

La valeur par défaut « window » place le lecteur au sommet de la pile d'affichage (aucun élément de la page ne peut être affiché au dessus). Les modes « opaque » et « transparent » doivent cependant être utilisés avec prudence, car ils peuvent provoquer des dysfonctionnements.

altContentId : optionnel - id du nœud dont on remplace le contenu par le lecteur FreepaperR (valeur par défaut, "freepaper1")

trace : optionnel – si true, affiche des infos sur l'avancée du traitement

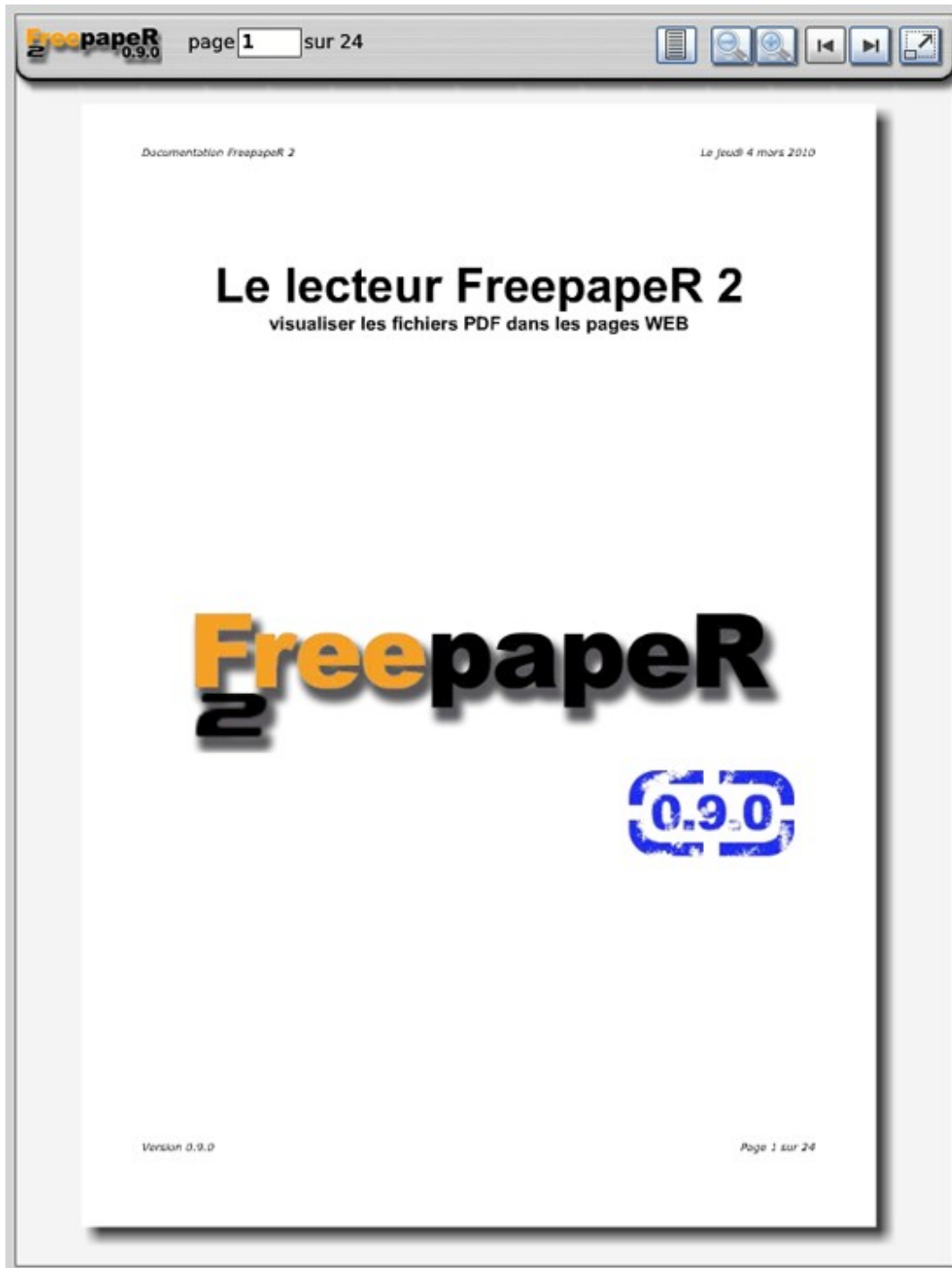
Si on souhaite ne pas définir une valeur, il suffit de ne pas écrire la paire correspondante. Si on ne souhaite définir aucune valeur pour un objet il faut tout de même créer un objet vide, par exemple :

attributes={}

ATTENTION : dans la description des attributs d'un objet (flashvars, params ou attributes), il est indispensable de séparer deux valeurs par le caractère virgule ",", sauf pour le dernier qui ne doit EN AUCUN CAS être suivi de virgule (cela provoque une erreur avec IE et Opéra) !

Lorsque la page décrite ci-dessus est affichée dans un navigateur , on obtient le lecteur **FreepaperR** avec le document à visualiser chargé :

Enfin presque !



Pour que tout se déroule comme précédemment décrit, il faut encore **installer la boîte à outils SWFTTOOLS** qui va prendre en charge la conversion du fichier PDF en SWF. L'installation est décrite en 3, **des solutions alternatives** sont proposées en 4.

3 Installer la boîte à outils swftools

Se rendre sur le site <http://www.swftools.org/download.html> et récupérer l'archive :

swftools-v.v.v.exe pour un système windows

swftools-yyyy-mm-dd-vvvv.tar.gz pour un système linux



Récupérer une archive d'une version 0.9 au minimum, sinon une conversion pour la machine virtuelle Flash 9 et supérieure (format AVM2) ne sera pas possible.

1. Windows

Lancer l'exécutable qui installe les utilitaires swftools. L'exécutable qui nous intéresse ici est « C:\Program Files\SWFTools\pdf2swf.exe ». Placer une copie dans le dossier de son choix.

2. Linux

Extraire les fichiers de l'archive, et les placer sur le serveur dans un dossier temporaire.

Se connecter par SSH au serveur (cela suppose d'avoir un accès SSH), se rendre dans le dossier temporaire où l'on a extrait les fichiers, puis lancer les commandes :

```
./configure (ayant auparavant réglé le bit d'exécution de ce fichier à 1)
```

Lorsque le traitement est terminé, lancer

```
make
```

On peut s'arrêter là, puis copier le binaire pdf2swf depuis le dossier src pour le placer dans le dossier de son choix. Bien penser à s'accorder les droits de lecture et d'exécution. On peut aussi si on a les droits **root** poursuivre l'installation dans le système avec **make install**.

3. Puis dans les deux cas

Si le binaire pdf2swf **n'est pas situé** dans le même dossier que le fichier html qui affiche le lecteur Freepaper, ouvrir le fichier pdf2swf.php et régler le contenu de la variable pdftoolsPath pour qu'elle indique le chemin vers l'exécutable pdf2swf, par exemple :

```
$this->pdftoolsPath='/kunden/homepages/16/d151613640/htdocs/bin/free/'; (Linux)
ou
$this->pdftoolsPath='D:\\www\\publications\\freepaperdemo\\'; (Windows)
ou encore
$this->pdftoolsPath='../'; (Linux)
ou bien encore
$this->pdftoolsPath='../\\'; (Windows)
```

Dans le dossier qui contient un PDF à visualiser, le système doit pouvoir créer un fichier avec l'extension `.swf` : il faut donc avoir les droits en lecture et en écriture dans ce dossier.

Remarque : La distribution est construite pour php version 4. Si votre serveur utilise php 5, alors vous pouvez renommer le fichier « `pdf2swf_php5.php` » en « `pdf2swf.php` ».

4 Je ne peux pas installer swftools sur mon serveur

Sans accès SSH, point de salut, on ne peut fabriquer le binaire `pdf2swf` (Linux).

Il y a des solutions alternatives

- J'ai trouvé un binaire pour ma distribution Linux

Dans ce cas, il suffit de placer ce binaire dans un dossier de son serveur, puis de régler la variable `$this->pdftoolsPath`, comme décrit en 3-3.

- Je télécharge le binaire en local, je converti les PDF en local, puis je place les fichiers sur le serveur

On procède pour l'installation de swftools comme décrit en 3-1 et 3-2.

Il faut ensuite convertir localement les fichiers PDF à visualiser.

1) Conversion en ligne de commande

Rem : on considère dans l'exemple que l'on est dans le même répertoire que le document à convertir

Lancer un shell, puis taper

```
<chemin/vers/pdf2swf>pdf2swf documentAVisualiser.pdf -o documentAVisualiser.swf -s internallinkfunction=handleInternalLink -T 9
```

sous Windows

ou

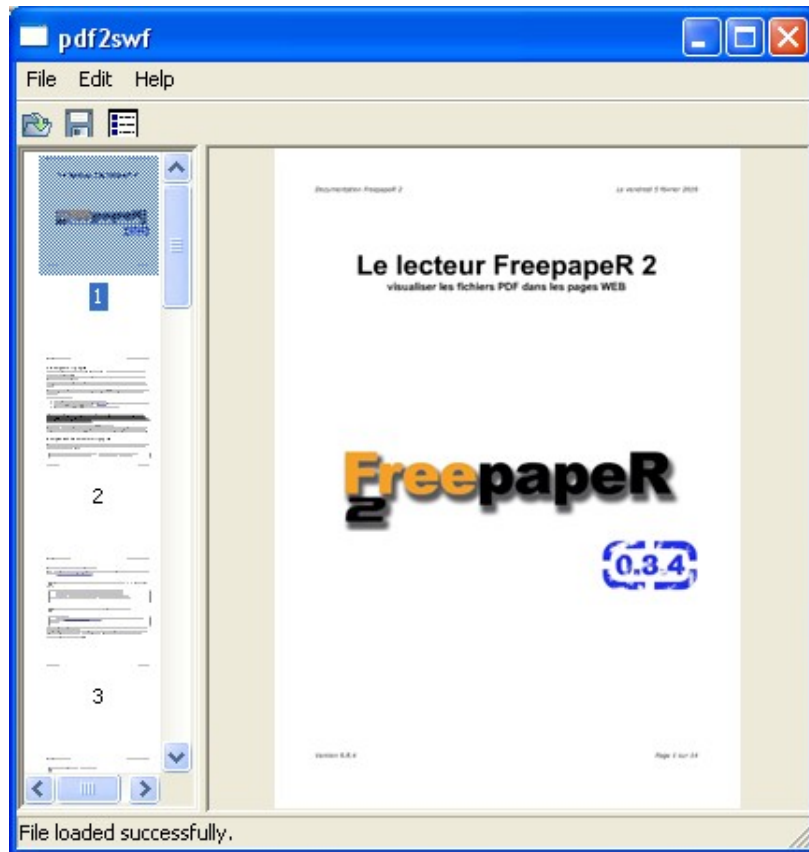
```
./pdf2swf documentAVisualiser.pdf -o documentAVisualiser.swf -s internallinkfunction=handleInternalLink -T 9
```

sous Linux

si le fichier PDF à convertir s'appelle `documentAVisualiser.pdf` par exemple.

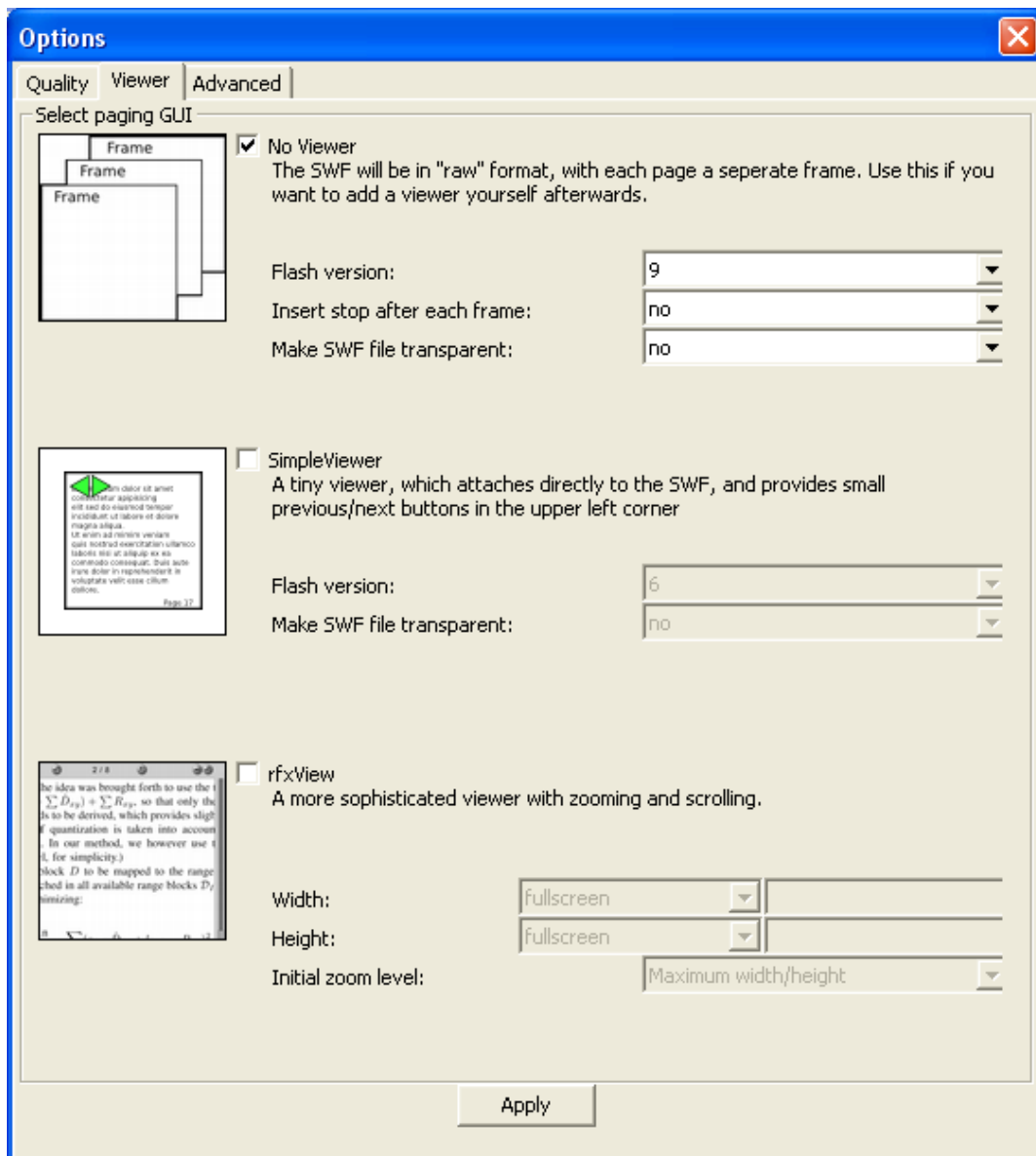
2) Utilisation de l'interface graphique (windows)

Lancer l'interface graphique pdf2swf, ouvrir le fichier à convertir (File/Open PDF).



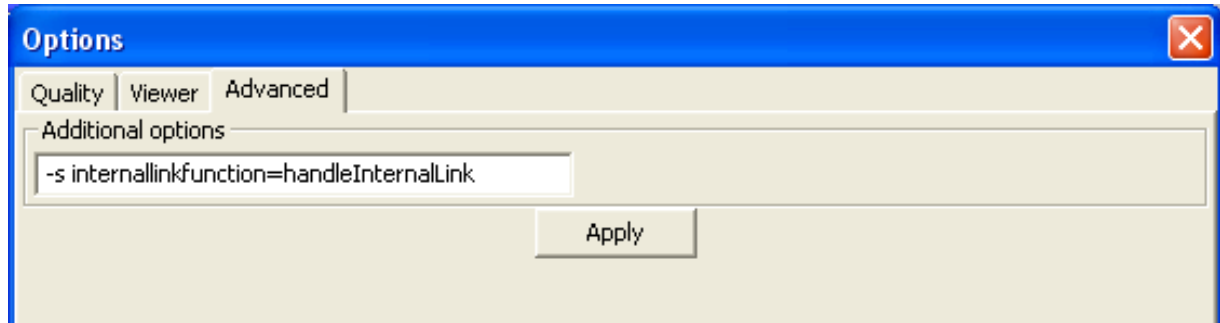
Choisir Edit/Options pour afficher la boîte de dialogue des options.

Cliquer sur l'onglet Viewer, cocher la case **No viewer** et choisir **9** dans la boîte déroulante Flash version.



Cliquer sur l'onglet Advanced et indiquer dans les options

-s internallinkfunction=handleInternalLink



Cliquer sur Apply.

Enfin, terminer par File/Save SWF (all pages). Indiquer dans la boîte de dialogue de sauvegarde le nom du dossier et du fichier à créer.

Cliquer sur Enregistrer.

3) Uploader le fichier sur le serveur

Utiliser votre client FTP favori pour copier le fichier swf généré manuellement sur le serveur,

⇒ On indique dans la variable docURL du fichier html le chemin du fichier swf que l'on a converti manuellement et placé sur le serveur, par exemple

```
//Options pour l'insertion d'un lecteur Freepaper
var flashvars = {
    docURL : "documents/gazetteCDM_2_200810.swf"      //obligatoire
};
```

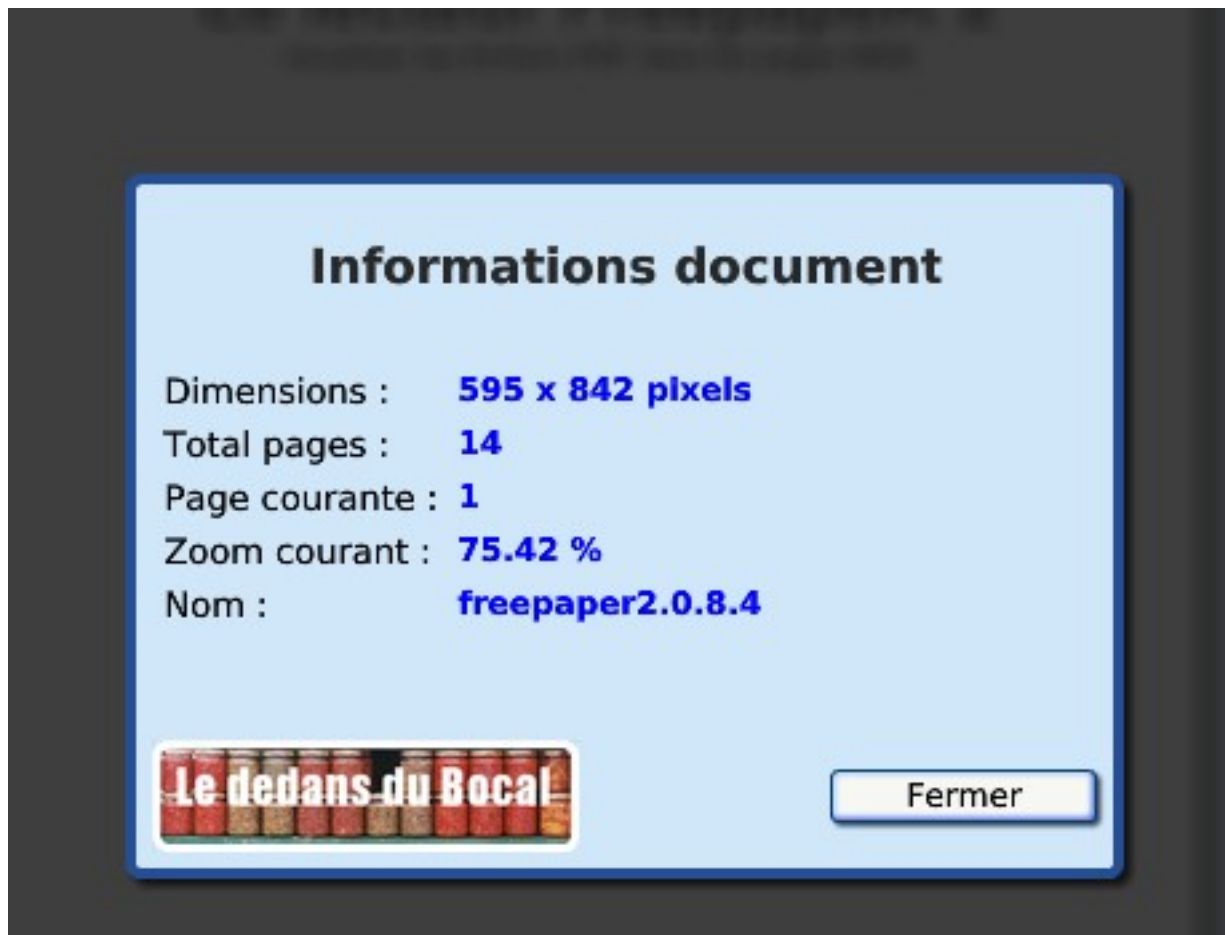
Aucune conversion n'étant requise en préambule à l'affichage, il n'est **pas fait appel à php** qui n'a donc pas besoin d'être installé sur le serveur.

5 Erreur rencontrées lors de la conversion

- | | |
|-------------------------|--|
| EXEC RETURN VALUE : 0 | ⇒ tout s'est bien déroulé |
| EXEC RETURN VALUE : 1 | ⇒ erreur lors de la conversion du fichier pdf |
| EXEC RETURN VALUE : 6 | ⇒ fichier PDF non lisible |
| EXEC RETURN VALUE : 11 | ⇒ erreur de segmentation (pdf2swf invalide) |
| EXEC RETURN VALUE : 126 | ⇒ pdf2swf a été trouvé mais n'est pas exécutable |
| EXEC RETURN VALUE : 127 | ⇒ le fichier pdf2swf n'a pas été trouvé |

6 Fenêtre d'information

Lorsque l'on clique sur le logo « Freepaper 2 » située en haut et à gauche de la barre de commande, on fait apparaître une fenêtre d'informations sur le document affiché :



Cette fenêtre disparaît dès que l'utilisateur clique sur le bouton « Fermer ».

7 Surcharge par un fichier de configuration

S'il existe dans le même dossier que la page HTML qui implante le lecteur Freepaper un fichier nommé « freepaper.xml », alors ce fichier de configuration est lu et les valeurs qu'il définit viennent surcharger les valeurs par défaut de comportement et d'affichage.

Il est aussi possible de spécifier un fichier de configuration situé dans un autre dossier et/ou ayant un nom différent. Il suffit pour cela de passer l'information (variable xmlDataPath) au script d'implantation de l'objet SWF :

```

var flashvars = {
    xmlDataPath : "<chemin vers le fichier de conf>/maConfig.xml",
    docURL : "nnnnnnnnnnnnnnn.ext"
};

```

On peut également placer des fichiers de localisation (différents langages) dans un sous dossier « language » situé dans le répertoire qui contient la page qui plante le lecteur Freepaper. De la sorte, en fonction de la langue de la machine virtuelle flash, le lecteur tente de lire le fichier suivant :

<code>langue.xml

Langue	Valeur
Tchèque	cs
Danois	da
Néerlandais	nl
Anglais	en
Finnois	fi
Français	fr
Allemand	de
Hongrois	hu
Italien	it
Japonais	ja
Coréen	ko
Norvégien	no
Autre/inconnu	xu
Polonais	pl
Portugais	pt
Russe	ru
Chinois simplifié	zh-CN
Espagnol	es
Suédois	sv
Chinois traditionnel	zh-TW
Turc	tr

Pour un visiteur russe, le lecteur tentera de charger le fichier de langue « language/ru.xml », pour un anglais « language/en.xml », pour un chinois simplifié « language/zh-CN.xml ». En cas d'échec, le français est utilisé comme langue par défaut.

Pour créer un fichier dans une nouvelle langue, il suffit de placer un fichier contenant les chaînes traduites dans le dossier language, Par exemple pour le tamoul, créer un fichier ta.xml

```
<!-- FreepaperR ressource file. English -->
<?xml version="1.0" encoding="UTF-8"?>
<ressourceStrings>
  <id_booklayout>தமிழ் 1</id_booklayout>
  <id_close>தமிழ் 2</id_close>
  <id_currentPage>தமிழ் 3 :</id_currentPage>
  <id_currentZoom>தமிழ் 4 :</id_currentZoom>
  <id_dimensions>தமிழ் 5 :</id_dimensions>
  <id_docInfo>தமிழ் 6</id_docInfo>
  <id_firstPage>தமிழ் 7</id_firstPage>
  <id_fitToPage>தமிழ் 8</id_fitToPage>
  <id_lastPage>தமிழ் 9</id_lastPage>
  <id_monoPageLayout>தமிழ் 10</id_monoPageLayout>
  <id_name>தமிழ் 11 :</id_name>
  <id_nextPage>தமிழ் 12</id_nextPage>
  <id_on>தமிழ் 13</id_on>
  <id_page>தமிழ் 14</id_page>
  <id_pixels>தமிழ் 15</id_pixels>
  <id_prevPage>தமிழ் 16</id_prevPage>
  <id_stackLayout>தமிழ் 17</id_stackLayout>
  <id_to>தமிழ் 18</id_to>
  <id_toFullScreen>தமிழ் 19</id_toFullScreen>
  <id_toStandardSize>தமிழ் 20</id_toStandardSize>
  <id_totalPages>தமிழ் 21 :</id_totalPages>
  <id_verticalListLayout>தமிழ் 22</id_verticalListLayout>
  <id_zoomMinus>தமிழ் 23</id_zoomMinus>
  <id_zoomPlus>தமிழ் 24</id_zoomPlus>
</ressourceStrings>
```

Il est aussi possible de spécifier un fichier de langue situé dans un autre dossier et/ou ayant un nom différent. Il suffit pour cela de passer l'information (variable langDataPath) au script d'implantation de l'objet SWF, par exemple :

```
var flashvars = {
    langDataPath : "<chemin vers le fichier de langue>/malangue.xml",
    docURL : "nnnnnnnnnnnnnnn.ext"
};
```

Voici un exemple d'implantation où l'on spécifie manuellement le fichier de langue et le fichier de configuration :

```
<script type="text/javascript">
//Options pour l'insertion d'un lecteur Freepaper
var flashvars = {
    langDataPath: "langues/ta.xml", //optionnel
    xmlDataPath : "xml/freepaperData.xml", //optionnel
    docURL : "documents/gazetteCDM_2_200810.pdf" //obligatoire
};
var params = {
    width:600, //optionnel
    height:800, //optionnel
    scale: "noScale", //optionnel
    wmode : "opaque", //optionnel
    allowFullScreen:"true" //obligatoire
};
var attributes = {
    altContentId:"freepaper1", //optionnel
    trace:true //optionnel
};
//Insertion d'un lecteur Freepaper
freepaper2.embedDoc(flashvars,params,attributes);
</script>
```

indique au système d'aller chercher le fichier de configuration « freepaperData.xml » dans le dossier « xml » et le fichier de localisation (langue) « ta.xml » dans le dossier « langues ».

Les attributs possibles du fichier XML (tous sont facultatifs) sont :

a) nœud racine du fichier XML

- couleur du fond : **backgroundColor**

Valeur par défaut : "0xDADADA"

- couleur de la bordure : **borderColor**

Valeur par défaut : "0x444444"

- épaisseur de la bordure : **borderWidth**

Valeur par défaut : "2" (pixels)

- affichage initial du document : **initialDisplay**

affichage initial du document : (F)it to page, page(W)idth, page(H)eight, % zoom <number>

Valeur par défaut : "F"

- mise en page initiale du document : **initialLayout**

mise en page initiale : "mono", "verticalList", "stack", "book"

Valeur par défaut : "mono"

b) nœud enfant de niveau 1 "<commandBar>"

- **leftImg**

nom complet de l'image pour la partie gauche de la barre de commande; Valeur par défaut: image métal gris avec arrondi (10x50 pixels)

- **rightImg**

nom complet de l'image pour la partie droite de la barre de commande; Valeur par défaut: image métal gris avec arrondi (10x50 pixels)

- **currentImg**

nom complet de l'image pour la partie courante de la barre de commande; Valeur par défaut: image métal gris (40x50 pixels)

- **horizAxis**

axe de symétrie horizontale sur la barre de commande servant à aligner les éléments: Valeur par défaut : 20pixels

c) nœud enfant de niveau 1 "<buttons>"

i. nœud enfant de niveau 2 "<fitToPage>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: flèche d'agrandissement sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: flèche de réduction sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: masque translucide (26x24 pixels)

ii. nœud enfant de niveau 2 "<nextPage>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image fin sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image fin enfoncée sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image fin sur fond gris contour orange (26x24 pixels)

iii. nœud enfant de niveau 2 "<prevPage>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image début sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image début enfoncée sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image début sur fond gris contour orange (26x24 pixels)

iv. nœud enfant de niveau 2 "<zoomPlus>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de "loupe avec +" sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de "loupe avec +" enfoncée sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de "loupe avec +" sur fond gris contour orange (26x24 pixels)

v. nœud enfant de niveau 2 "<zoomMinus>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de "loupe avec -" sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de "loupe avec -" enfoncée sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de "loupe avec -" sur fond gris contour orange (26x24 pixels)

vi. nœud enfant de niveau 2 "<monoPageLayout>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de page sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de page enfoncé sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de page sur fond gris contour orange (26x24 pixels)

vii. nœud enfant de niveau 2 "<verticalListLayout>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de 2 pages superposées sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de 2 pages superposées enfoncé sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de 2 pages superposées sur fond gris contour orange (26x24 pixels)

viii. nœud enfant de niveau 2 "<stackLayout>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de 2 pages juxtaposées sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de 2 pages juxtaposées enfoncé sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de 2 pages juxtaposées sur fond gris contour orange (26x24 pixels)

ix. nœud enfant de niveau 2 "<bookLayout>"

- **UpImg**

nom complet de l'image pour l'état "haut"; Valeur par défaut: image de page en rotation sur fond gris (26x24 pixels)

- **downImg**

nom complet de l'image pour l'état "bas"; Valeur par défaut: image de page en rotation enfoncée sur fond gris (26x24 pixels)

- **overImg**

nom complet de l'image pour l'état "survolé"; Valeur par défaut: image de page en rotation sur fond gris (26x24 pixels)

Pour ne pas modifier la valeur par défaut d'un des attributs, il suffit de l'omettre ou de régler sa valeur à "".

ASTUCE !

Pour surcharger les images qui définissent un bouton, il faut au moins définir l'attribut **upImg**. En effet, il est utilisé pour définir la zone de clic du bouton. Toutes les images du bouton dont l'attribut n'est pas défini dans le fichier XML seront dessinées avec l'image définie pour l'attribut **upImg**. Si **upImg** n'est pas défini, alors les images du bouton par défaut seront utilisées.

Pour « cacher » un bouton, il suffit de définir uniquement l'attribut **upImg** et de lui donner une valeur d'image inexistante, par exemple " " (caractère espace).

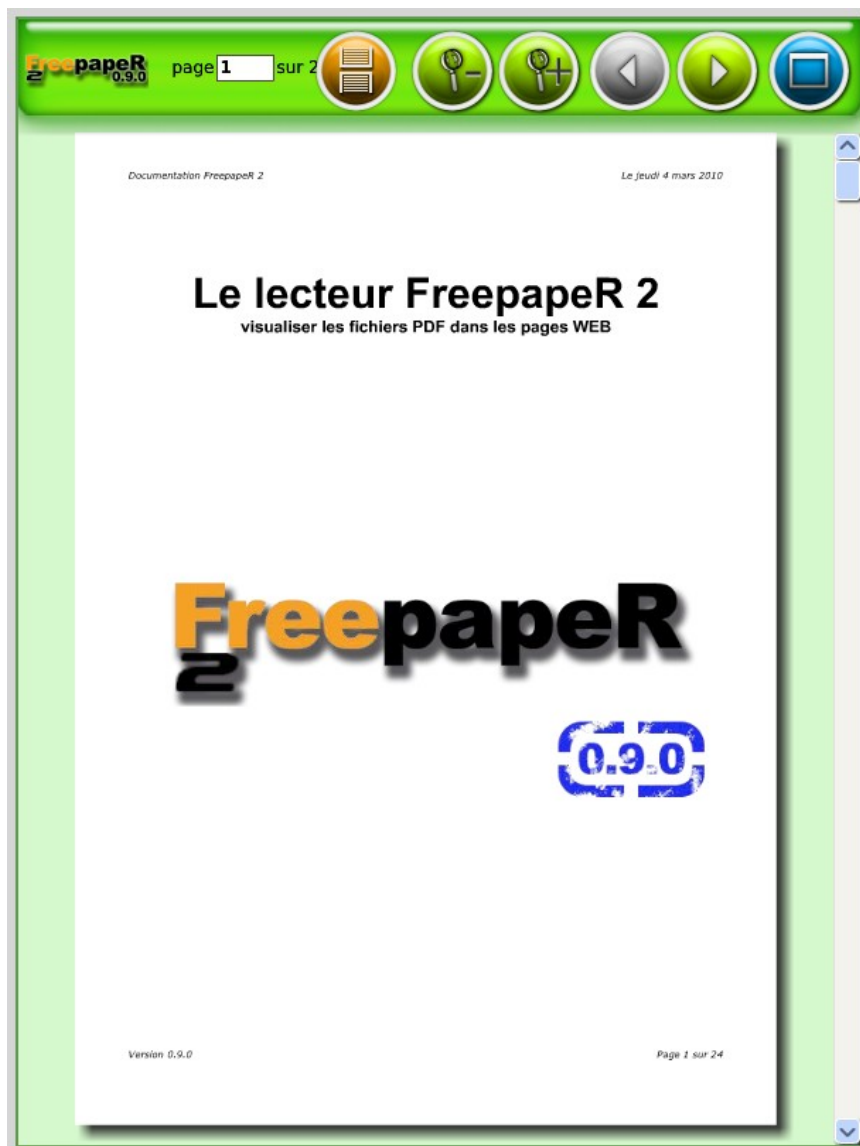
ASTUCE 2 !

Pour ne pas proposer une des mises en page possibles (par exemple « liste verticale »), il suffit de « cacher » le bouton correspondant (voir astuce 1).

8 L'exemple contenu dans l'archive

Avec les éléments fournis dans le fichier freepaper2.0.9.0.zip, on a le choix de 2 habillages pour surcharger l'habillage par défaut, et on obtient les interfaces suivantes :





Les fichiers de configuration xml respectifs sont dans les dossiers « images/softBlue » et « images/glossyGreen ».

9 Les nouveautés

De la version 0.9.0 :

- On propose une nouvelle mise en page, « Livre » qui simule le changement de page avec une animation rappelant le mouvement des pages d'un livre papier.
- Des infobulles sont rajoutées sur les boutons de commande.
- Tous les chaînes de langue de l'interface peuvent être localisées et il est possible de changer leur valeur simplement en modifiant ou en créant un fichier xml.

De la version 0.8.4 :

- Le lecteur fonctionne maintenant dans la machine virtuelle AVM2 apparue avec le lecteur flash 9. Cependant, il reste en mesure d'afficher les documents swf générés en AVM1 (avant flash 9).
- Le document est placé dans un Panneau disposant d'ascenseurs s'il devient trop grand pour tenir dans la vue. On peut déplacer le document avec la roulette de la souris.
- Les touches « Début », « Fin », « Page précédente », « Page suivante », « Flèche bas », « Flèche haut », « Flèche gauche » et « Flèche droite » permettent de se déplacer dans le document (suivant le type d'affichage), sauf en mode plein écran (clavier non géré).
- Le lecteur dispose maintenant de 3 modes d'affichage : « Page simple » (comme auparavant), « Liste de pages » et « Pile ».

De la version 0.8.3 :

- Remplacement du champ de sélection de la page à afficher par un composant disposant aussi d'un curseur que l'on peut déplacer à la souris, permettant ainsi la navigation dans le document même en mode plein écran (le clavier est désactivé dans les objets swf en mode plein écran).
- Ajout des éléments « Première page », « Page précédente », « Page suivante » et « Dernière page » dans le menu contextuel situé sur le document affiché (clic droit de la souris).

De la version 0.8.1 :

- Détection du système du serveur. Il n'y a plus besoin d'intervenir sur la valeur de la variable « \$this->pdftoolsPath » du fichier « php/pdf2swf.php » si le binaire pdf2swf est situé dans le même dossier que le fichier index.html.
- Ajout du paramètre wmode pour l'insertion du lecteur Freepaper. Une valeur « opaque » ou « transparent » permet de le replacer dans le système de couche du DOM (ce qui l'autorise à être affiché en dessous d'autres éléments HTML). La valeur par défaut « window » place le lecteur au sommet de la pile d'affichage (aucun élément de la page ne peut être affiché au dessus). Les modes « opaque » et « transparent » doivent cependant être utilisés avec prudence, car ils peuvent provoquer des dysfonctionnements.

De la version 0.8.0 :

- Lors du glissé du document, il n'est plus possible de faire glisser la page hors des limites du lecteur
- Utilisation de la molette de la souris pour faire défiler la page
- Modification du mode plein écran : affichage sur la totalité de l'écran.



De la version 0.7.0

- Ouverture du document selon un des 4 modes suivants:
 - ajusté à la page, ajusté à la hauteur du lecteur, ajusté à la largeur du lecteur, valeur de zoom (%)
- Ajout d'une fenêtre d'informations sur le document
- Lors d'un changement de page, le haut de la page est re-positionné juste sous la barre de commande

- Personnalisation possible par fichier XML :
 - de la couleur du fond du lecteur
 - de la couleur du contour du lecteur
 - de l'épaisseur du contour du lecteur
 - des 3 images qui composent la barre de commande
 - de la position de l'axe d'alignement vertical des éléments de la barre de commande
 - des 5 boutons (3 images possibles pour chaque) de la barre de commande
 - du mode d'ouverture document

De la version 0.6.0

- La fonction Zoom a été améliorée : le zoom est maintenant effectué par rapport au point situé au centre de la visionneuse
- Ajout de la fonctionnalité de visualisation « pleine page » (la visionneuse occupe tout l'espace disponible dans le navigateur)