

Le Plugin SPIP FreepapeR 2

visualiser les fichiers PDF dans les pages WEB

FreepapeR



Toutes versions de SPIP à partir de SPIP 1.9

1 Pourquoi FreepapeR

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 inséré.

FreepapeR 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 programme SWF 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 FreepapeR).

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 plugin

Extraire le contenu de l'archive freepaper2-spip.0.9.0.zip, dans le dossier plugins de votre installation SPIP.

OU

créer un sous-dossier /plugins/auto pour activer l'installation automatique de plugins. Il suffit ensuite d'indiquer l'URL du fichier zip du plugin et de suivre les indications du système (l'installation automatique est disponible à partir de SPIP 2.0).

Pour le plugin freepaper2-spip.0.9.0, l'URL à indiquer est :

```
http://lede dansdubocal.net/IMG/zip/freepaper2-spip.0.9.0.zip
```

L'activer via l'administration des plugins. (Consulter la [documentation officielle](#) pour plus de détails.)

Rappel : pour charger correctement les fichiers nécessaire au fonctionnement du plugin s'assurer que la balise #INSERT_HEAD est incluse dans la partie <head> du squelette (la ligne <INCLUDE {fond=inc-head}>, que l'on trouve dans les squelettes de la distribution se charge de cela).

Dans ce plugin, on utilise l'utilitaire javascript d'installation d'objet Flash **swfobject 2.2**, disponible à l'adresse suivante : <http://code.google.com/p/swfobject/>

En conséquence, il conviendra d'intégrer ce fichier dans la partie « head » des pages qui doivent utiliser le plugin:

```
<script type="text/javascript" src="#CHEMIN{javascript/swfobject.js}"></script>
```

par exemple, si la librairie swfobject est située dans le répertoire « javascript » de votre installation SPIP.

OU

utiliser le plugin SPIP SWFObject 2.2, qui installe la librairie swfobject.js, version 2.2 dans l'espace public et privé, disponible à l'adresse <http://lede dansdubocal.net/spip.php?article19>

OU

Installer la variante « Installation du plugin SPIP FreepaperR 2 0.9.0 avec swfobject » qui installe le plugin FreepaperR 2 avec une version spécifique de swfobject, qui n'écrase pas les éventuelles autres version installées de swfobject (l'installation automatique via le répertoire plugins/auto est là aussi possible).

3 La balise #FPP2STD

Lorsque le plugin est activé, on dispose de la balise #FPP2STD qui liste tous les documents « pdf » joints à l'article et intègre pour chacun d'eux un lecteur FreepaperR dans la page, avec les valeurs par défaut.

Les paramètres suivant permettent de modifier le comportement par défaut :

```
#FPP2STD{xmlData=nomFichierXml} nom du fichier xml de configuration (situé dans le
sous-répertoire xml du plugin), nom par défaut : freepaper.xml
#FPP2STD{xmlLang=NomDeFichier} chemin du fichier xml de localisation (dans le dossier
langage du plugin), valeur par défaut : <langueMachineVirtuelle>.xml (en.xml, zh-
CN.xml, ...)
#FPP2STD{largeur=nbPixels} largeur de la page FreepaperR. Défaut : 600
#FPP2STD{hauteur=nbPixels} hauteur de la page FreepaperR. Défaut : 800
#FPP2STD{trace=true ou false} activation du mode verbeux. Défaut : false
#FPP2STD{wmode=window ou opaque ou transparent} fixe le mode fenêtré de l'objet swf
pour la transparence et le placement dans les couches du navigateur. Défaut : window
```

Ainsi

```
[ (#FPP2STD{largeur=300}{hauteur=400}{trace=true}) ]
```

implante un lecteur FreepaperR de largeur 300px, de hauteur 400px avec le mode trace activé (utile en phase de création des squelettes pour déceler les problèmes éventuels). Si le fichier xml/freepaper.xml existe, alors les paramètres qu'il définit sont utilisés pour modifier l'aspect/le comportement du lecteur.

```
[ (#FPP2STD{xmlData=freepaperSample.xml}{largeur=300}{hauteur=400}
{trace=true}) ]
```

implante le même lecteur que précédemment mais en utilisant les définitions du fichier de configuration xml/freepaperSample.xml

On peut aussi utiliser une notation alternative :

```
[ (#FPP2STD{xmlData=freepaperSample.xml,largeur=300,hauteur=400}) ]
```

par exemple, où les paramètres sont passés en une seule fois avec comme séparateur la virgule.

4 La balise #FPP2SWF

Possède exactement la même syntaxe que la balise #FPP2STD, mais elle implante tous les documents « swf » joints à l'article.

Aucune conversion n'étant requise en préambule à l'affichage du document, il n'est pas fait appel à la librairie php qui n'a donc pas besoin d'être installé sur le serveur. Il faudra par contre disposer de documents swf (récupérés par ailleurs ou convertis localement) qu'il faudra copier sur le serveur.

5 Le modèle modelefppII (2 en chiffre romain)

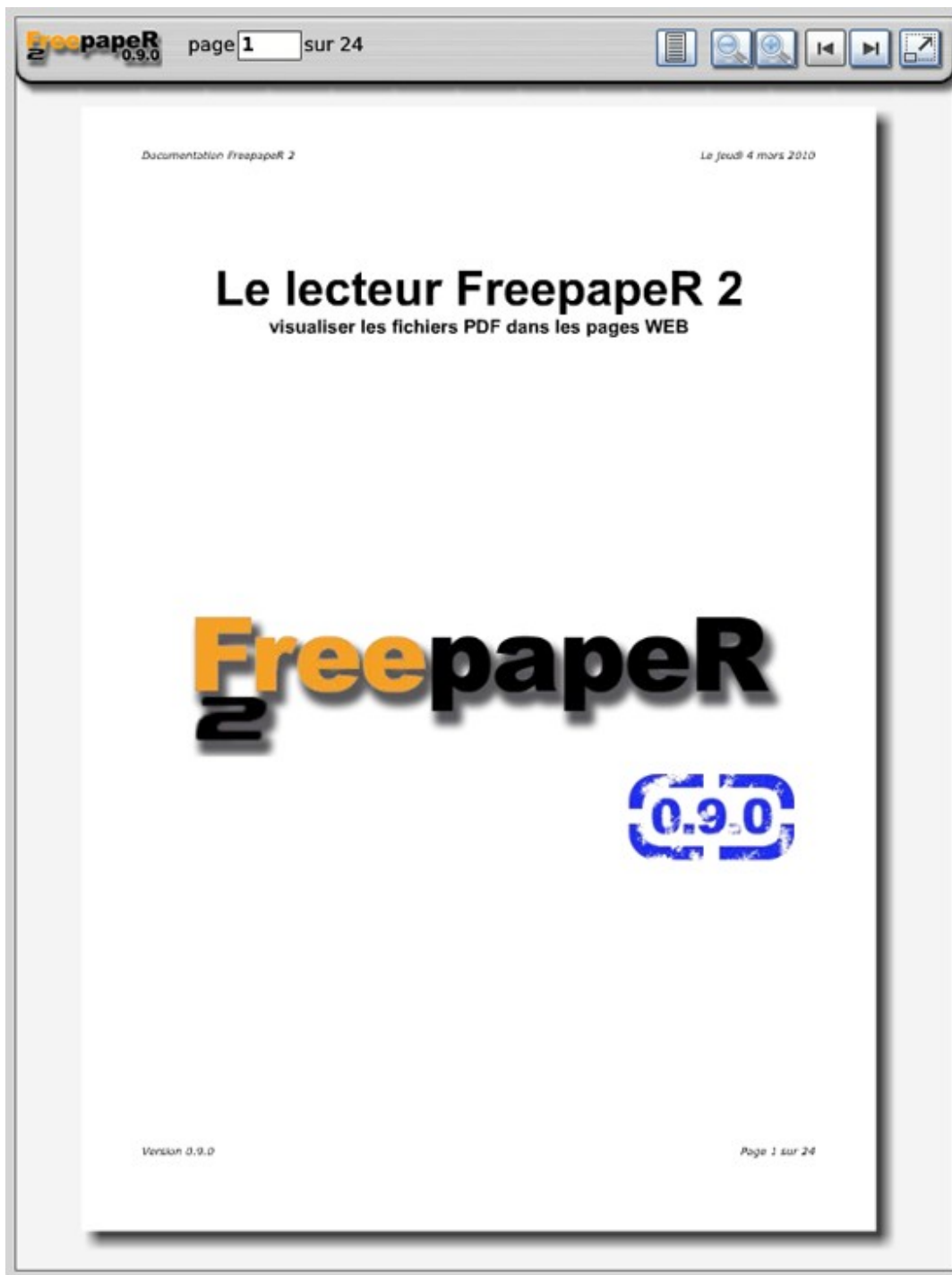
Lorsque le plugin est activé, il permet l'utilisation du modèle modelefppII dans le corps des articles.

La syntaxe complète est :

```
<modelefppIIxxx                                ou xxx est l'id du document PDF à visualiser
|xmlData=nomFichierXml                         -> par défaut freepaper.xml
|xmlLang=nomFichierXmlLcalisation              -> par défaut <langueMachineVirtuelle>.xml
|hauteur=nbPixels                              -> par défaut 600
|largeur=nbPixels                              -> par défaut 800
|trace=true ou false                           -> par défaut false
|wmode=window ou opaque ou transparent         -> par défaut window
>
```

Remarque : Dans l'espace d'administration, les lecteurs FreepapeR implantés par modèle sont représentés par l'image « Get FLASH PLAYER ». Cela est normal. On visualise ainsi qu'il y a à cet endroit de l'article un objet implanté, mais voir son contenu n'a pas d'intérêt ici.

On obtient dans la page publique du site un lecteur **FreepapeR** 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 6, **des solutions alternatives** sont proposées en 7.

6 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-xxxx.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

Extraire les fichiers de l'archive, puis placer l'exécutable pdf2swf.exe sur le serveur, à la racine du dossier du plugin FreepapeR.

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 à la racine du dossier du plugin FreepapeR. Bien penser à s'accorder les droits de lecture et d'exécution sur ce fichier.

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 ».

7 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 à la racine du dossier du plugin FreepapeR.

- Je télécharge le binaire en local, je converti les fichiers PDF en local, puis je place les fichiers obtenus sur le serveur qui héberge mon site SPIP.

On procède pour l'installation de swftools comme décrit en 6-1 et 6-2.

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

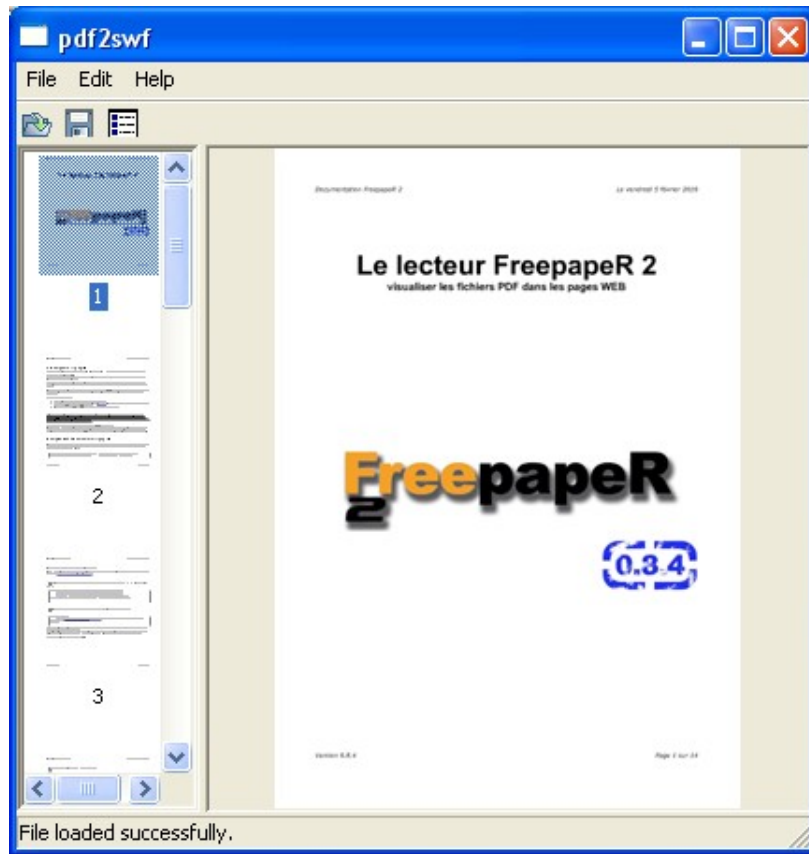
```
./pdf2swf documentAVisualiser.pdf -o documentAVisualiser.swf -s  
internallinkfunction=handleInternalLink -T 9
```

sous Linux

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

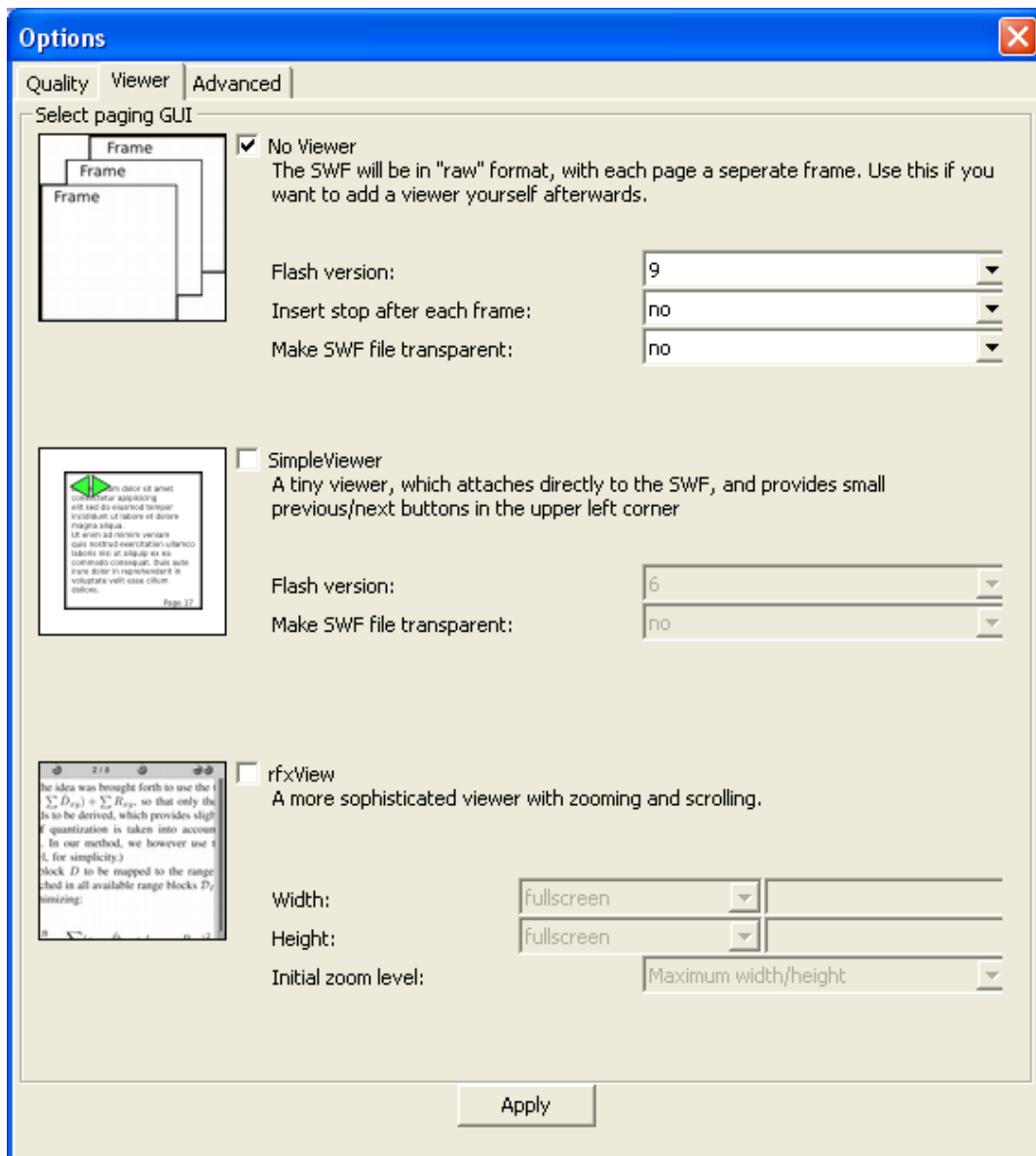
2) Utilisation de l'interface graphique (windows)

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



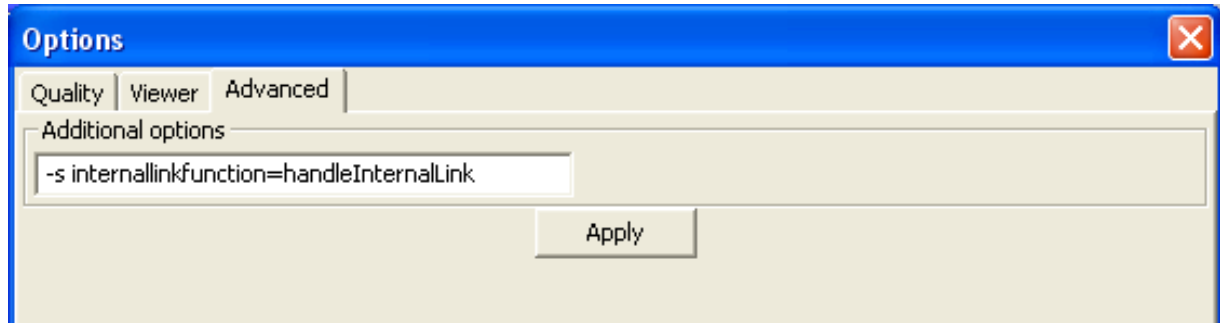
Choisir Edit/Options pour afficher la boîte de dialogue des options.

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

Joindre le document swf que l'on vient de générer à l'article SPIP souhaité.

8 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

9 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 ».

10 Surcharge par un fichier de configuration

Dans le répertoire du plugin FreepaperR, il existe un sous-répertoire « xml » dans lequel se situe un fichier nommé «freepaper.xml ». Ce fichier de configuration est lu et les valeurs qu'il définit viennent surcharger les valeurs par défaut de comportement.

Il est aussi possible de spécifier un fichier de configuration ayant un nom différent. Il suffit pour cela de passer l'information (variable xmlData des balises ou paramètre xmlData du modèle) lors de l'implantation.

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 horizontal 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. De plus, seules ses zones opaques sont considérées comme actives. 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).

11 Localisation (langue) de l'interface

On peut placer des fichiers de localisation (différents langages) dans le sous-répertoire « language » du plugin. De la sorte, en fonction de la langue de la machine virtuelle flash, le lecteur tente de lire le fichier <code>langue.xml</code> suivant :

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.

Dans la distribution, seul le fichier pour la langue anglaise est fourni,

Pour créer un fichier dans une nouvelle langue, il suffit de placer un fichier contenant les chaînes traduites dans le sous-répertoire « 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>

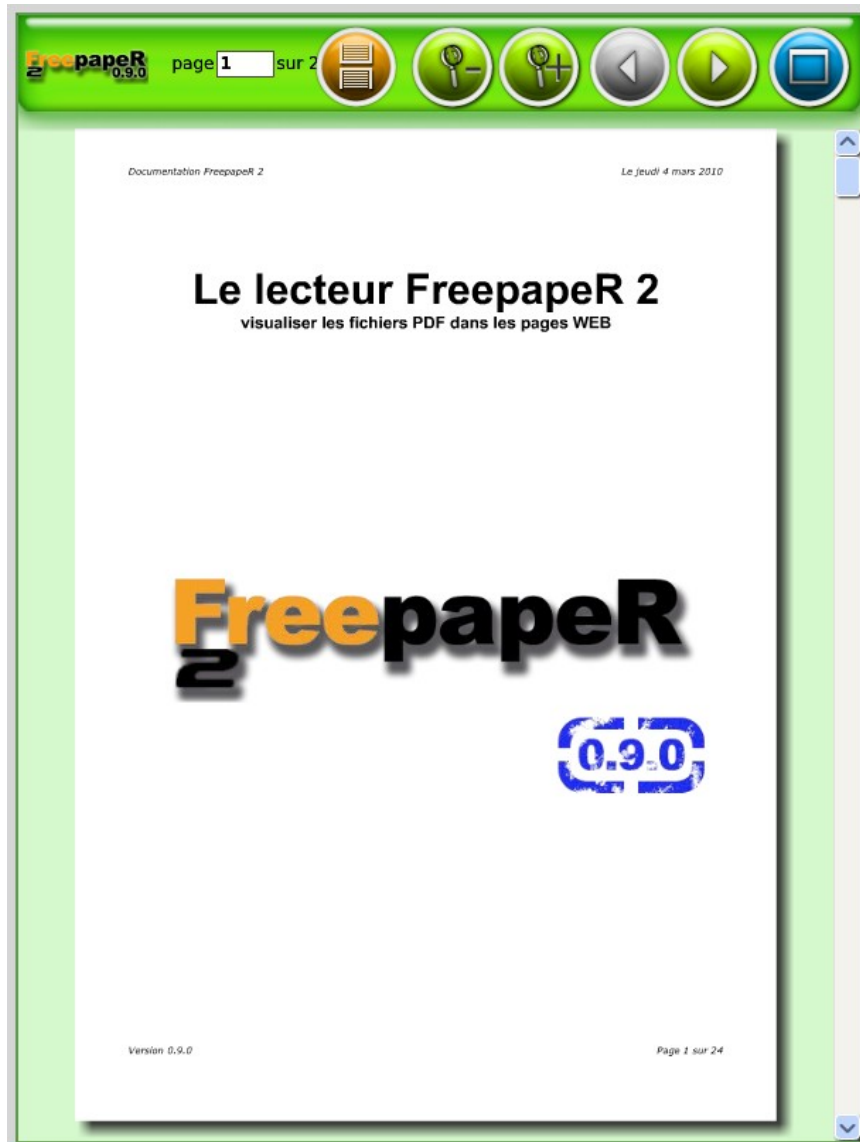
```

Il est aussi possible de spécifier un fichier de configuration ayant un nom différent. Il suffit pour cela de passer l'information (variable xmlLang des balises ou paramètre xmlLang du modèle) lors de l'implantation.

12 L'exemple contenu dans l'archive

Avec les éléments fournis dans le fichier freepaper2-spip.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 ».

13 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 :

- La version 0.8.3 fonctionne avec toutes les versions de SPIP supérieures à 1.9 (donc y compris pour les versions SPIP à partir de 2.0.9)
- 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 0.8.2 :

- La version 0.8.2 apporte comme seule nouveauté la compatibilité avec les versions de SPIP supérieures à 2.0.9. Pour une utilisation avec les versions antérieures de SPIP, utiliser le plugin FreepapeR v 0.8.1.

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 »

- 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 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.
- Le plugin est maintenant compatible avec le système d'installation automatique (répertoire plugins/auto) disponible depuis la version SPIP 2.0.

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)