RECAP DES FONCTIONS TCPDF

```php
<?php
//Chargement de la bibliothèque TCPDF
require_once('../TCPDF-main/examples/tcpdf_include.php');

//Création de l'objet PDF
$pdf = new TCPDF(PDF_PAGE_ORIENTATION, PDF_UNIT, PDF_PAGE_FORMAT, true, 'UTF-8', false);

//Informations sur le document
$pdf->setCreator('Jean Dupont');
$pdf->setAuthor('Frédéric SOL');
$pdf->setTitle('Tutoriel TCPDF');
$pdf->setSubject('Recap des méthode de TCPDF');
$pdf->setKeywords('TCPDF, PDF, Exemple, fonctions');

/***************
*    HEADER    *
***************/

/*
La méthode SETHEADERDATA prend en paramètre 6 arguments :
1er argument (définit dans le fichier tcpdf_autoconfig.php) : le nom du fichier du logo en haut à gauche
2e argument (définit dans le fichier tcpdf_autoconfig.php) : un nombre pour la taille du logo en haut à gauche
3e argument (définit dans le fichier tcpdf_config.php) : le titre de l'entête
4e argument (définit dans le fichier tcpdf_config.php) : le sous-titre de l'entête
5e argument : la couleur du texte de l'entête
6e argument : la couleur de la ligne qui est sous le texte d'entête
*/
$pdf->setHeaderData(PDF_HEADER_LOGO, PDF_HEADER_LOGO_WIDTH, PDF_HEADER_TITLE.'001', PDF_HEADER_STRING_SOUS_TITRE, array(0,64,255), array(187,11,11));

/*
La méthode SETHEADERFONT prend en paramètre un tableau contenant 3 éléments :
1er argument (définit dans le fichier tcpdf_config.php) : Le nom de la police
2e argument (définit dans le fichier tcpdf_config.php) : le style (N pour normal, I pour italic, B pour bold, U pour underline)
3e argument (définit dans le fichier tcpdf_config.php) : La taille du texte
*/
$pdf->setHeaderFont(Array(PDF_FONT_NAME_MAIN, 'I', PDF_FONT_SIZE_MAIN));

/*La méthode SETHEADERMARGIN prend 1 paramètre un argument
1er argument (définit dans le fichier tcpdf_config.php) : un nombre pour la marge en haut
```

```php
*/
$pdf->setHeaderMargin(PDF_MARGIN_HEADER);

//Suppression de l'entête
$pdf->setPrintHeader(false);

/**************
*    FOOTER    *
***************/

/*
La méthode SETFOOTERDATA prend en paramètre 2 arguments :
1er argument : La couleur du texte du footer
2e argument : La couleur de la ligne qui est au-dessus du texte du footer
*/
$pdf->setFooterData(array(187,11,11),array(0,64,255));

/*
La méthode SETFOOTERFONT prend en paramètre un tableau contenant 3 éléments :
1er argument (définit dans le fichier tcpdf_config.php) : Le nom de la police
2e argument (définit dans le fichier tcpdf_config.php) : Le style de la police
(N pour normal, I pour italic, B pour bold, U pour underline)
3e argument (définit dans le fichier tcpdf_config.php) : La taille du texte
*/
$pdf->setFooterFont(Array(PDF_FONT_NAME_DATA, 'I', PDF_FONT_SIZE_DATA));

/*La méthode SETFOOTERMARGIN prend 1 paramètre un argument
1er argument (définit dans le fichier tcpdf_config.php) : un nombre pour la
marge en bas
*/
$pdf->setFooterMargin(PDF_MARGIN_FOOTER);

/*
La méthode SETMARGINS prend en paramètre 3 arguments :
1er paramètre (définit dans le fichier tcpdf_config.php) : la marge à gauche
2e paramètre (définit dans le fichier tcpdf_config.php) : la marge en haut
après l'entête. Si on augmente trop la marge du haut il faudra augmenter la
valeur de PDF_MARGIN_HEADER
3e paramètre (définit dans le fichier tcpdf_config.php) : la marge à droite
*/
$pdf->setMargins(PDF_MARGIN_LEFT, PDF_MARGIN_TOP, PDF_MARGIN_RIGHT);

//Suppression du footer
$pdf->setPrintFooter(false);


/*************************
*    POLICE ET COULEUR    *
*************************/
```

```php
/*
La méthode SETFONT prend en paramètre 6 arguments :
1er argument : Le nom de la police
2e argument : Le style de la police (N pour normal, I pour italic, B pour
bold, U pour underline)
3e argument : La taille de la police
4e argument : Le fichier de police à insérer
5e argument : ???
6e argument : ???
*/
$pdf->setFont('dejavusans', 'N', 14);

//Couleur du texte
$pdf->setTextColor(0, 63, 127);

//couleur de fond de la prochaine cellule
$pdf->setFillColor(255, 255, 127);


/**************************
*    CREATION DES PAGES    *
**************************/

/*
La méthode ADDPAGE prend en paramètre 4 arguments :
1er argument : L'orientation de la apge :
    P : Portrait
    L : Paysage
2e argument : Le format de page (A4, A3, ...)
3e argument : Garder ou pas les marges
4e argument : ???
*/

$pdf->AddPage('P', 'A4');

/*
La méthode SETAUTOPAGEBREAK prend en paramètre 2 arguments :
1er argument : Activer ou non le saut de page
2e argument : La marge en bas
*/
$pdf->setAutoPageBreak(TRUE, PDF_MARGIN_BOTTOM);

//Copie de la 1ère page
$pdf->copyPage(1);

/**************************
*    ECRITURE DU TEXTE    *
**************************/
```

```
/*La méthode WRITE prend en paramètres 11 arguments
1er argument : La marge en haut
2e argument : Le texte à afficher
3e argument : lien hypertexte sur le bloc
4e argument : Appliquer ou non couleur de fond (définit auparavant avec la
méthode setFillColor)
5e argument : Alignement du texte
     L : à gauche (par défaut)
     C : centré
     R : à droite
6e argument : Si vrai le curseur sera en fin de ligne, sinon il sera en début
de ligne suivante
7e argument : stretch
    0 : Désactivé
    1 : Mise à l'échelle horizontale uniquement si le texte est plus grand que
la largeur de la cellule
    2 : Mise à l'échelle horizontale forcée pour s'adapter à la largeur de la
cellule
    3 : Espacement des caractères uniquement si le texte est plus grand que la
largeur de la cellule
    4 : Espacement forcé des caractères pour s'adapter à la largeur de la
cellule
8e argument : Si vrai imprime uniquement la première ligne et renvoie la
chaîne restante.
9e argument : Si vrai la chaîne est le début d'une ligne
10e argument : Hauteur max
11e argument : La largeur de la première ligne sera réduite de ce montant
12e argument : Tableau margin du conteneur parent
*/
$pdf->write(100,"lorem ipsums ade amgfdg fdg ",'www.fred-
sol.fr',false,'C',false,0,true,false,200,'','');

/*
La méthode WRITEHTML prend en paramètres 6 arguments
1er argument : Le texte HTML à afficher
2e argument : Effectuer ou non un saut de ligne à la fin de ce contenu
3e argument : Appliquer ou non couleur de fond (définit auparavant avec la
méthode setFillColor)
4e argument : Réinitialise ou non la hauteur de ligne
5e argument : Auto padding
6e argument : Alignement du texte
     L : à gauche (par défaut)
     C : centré
     R : à droite
*/
$pdf->writeHTML($html2,true,true,false,true,'R');

/*
```

```
La méthode WRITEHTMLCELL permet d'écrire dans une cellule avec la possibilité
de mettre des bordure, une couleur de fond et du text HTML (tous les attributs
HTML doivent être entre doubles quotes).
Cette méthode prend en paramètre plusieurs arguments :
1er argument : la longueur de la cellule. Si on met 0 alors la cellule
occupera toute la largeur (jusqu'à la limite de la marge)
2e argument : la hauteur de la cellule
3e argument : la coordonnée X (Si X est renseigné alors l'argument de la
position de la prochaine cellule ne changera rien à la position de la cellule)
4e paramètre : la coordonnée Y (Si Y est renseigné alors l' argument de la
position de la prochaine cellule ne changera rien à la position de la cellule)
5e argument : le texte HTML à afficher
6e argument : la bordure :
    0 : Pas de bordure
    1 : bordure de chaque coté
    XXXX :
        T pour une bordure en haut
        R pour une bordure à droite
        B pour une bordure en bas
        L pour une bordure à Gauche
            Exemple : LR pour une bordure à gauche et  àdroite
    Array : Pour définir le type de bordure, l'épaisseur, la couleur, ...
        Exemple : array('LTRB' => array('width' => 2, 'cap' => 'butt', 'join'
=> 'miter', 'dash' => 0, 'color' => array(0, 0, 0)))
7e argument : Position du prochain élément
    0: à droite
    1: au début de la ligne du dessous
    2: en dessous et décalé de la longueur de la cellule précédente
8e argument : Appliquer ou non une couleur de fond (définit auparavant avec la
méthode setFillColor)
9e argument : Reinitialiser la hauteur de la dernière cellule
10e argument :  Alignement du texte
     L : à gauche (par défaut)
     C : centré
     R : à droite
11e argument : Ajouter ou non un padding en haut et en bas
*/
$pdf->writeHTMLCell(0, 0, 50, 50, $html1, 0, 1, 0, true, '', true);

/*
La méthode SETCELLPADDINGS permet de définir le padding des prochaines
cellules. Elle prend en paramètre 4 arguments :
1er argument : padding à gauche
2e argument : padding en haut
3e argument : padding à droite
4e argument : padding en bas
*/
$pdf->setCellPaddings(1, 1, 1, 1);
```

```
/*
La méthode SETCELLMARGINS permet de définir le margin des prochaines cellules.
Elle prend en paramètre 4 arguments :
1er argument : padding à gauche
2e argument : padding en haut
3e argument : padding à droite
4e argument : padding en bas
*/
//margin de la prochaine cellule
$pdf->setCellMargins(1, 1, 1, 1);

/*
La méthode CELL prend en arguments 13 arguments :
1er argument : La longueur de la cellule
2e argument : La hauteur de la cellule
3e argument : Le texte à afficher
4e argument : Une bordure (voir la méthode WRITEHTMLCELL)
5e argument : Position du prochain élément.
    0: à droite
    1: au début de la ligne du dessous
    2: en dessous et décalé de la longueur du bloc
6e argument : Alignement du texte
     L : à gauche (par défaut)
     C : centré
     R : à droite
7e argument :  Appliquer ou non couleur de fond (définit auparavant avec la méthode setFillColor)
8e argument : L'URL
9e argument : stretch mode:
    0 : Désactivé
    1 : Mise à l'échelle horizontale uniquement si le texte est plus grand que la largeur de la cellule
    2 : Mise à l'échelle horizontale forcée pour s'adapter à la largeur de la cellule
    3 : Espacement des caractères uniquement si le texte est plus grand que la largeur de la cellule
    4 : Espacement forcé des caractères pour s'adapter à la largeur de la cellule
10e argument : Ignorer ou non la hauteur minimale
11e argument : Alignement de la cellule par rapport à l'axe Y de la page
12e argument : Alignement vertical (fonctionne que si le 11e argument (texteHTML) = false)
    T : En haut
    M : Au milieu
    B : En bas
*/
$pdf->Cell(100, 200, 'monTexte', 1, 0, 'C', true, 'http://www.fred-sol.fr', 0, false, 'T', 'M')
```

```
/*
La méthode MULTICELL prend en arguments 16 arguments :
1er argument : La longueur de la cellule
2e argument : La hauteur de la cellule
3e argument : Le texte à afficher
4e argument : Une bordure
5e argument : Alignement du texte
     L : à gauche (par défaut)
     C : centré
     R : à droite
6e argument :  Appliquer ou non couleur de fond (définit auparavant avec la
méthode setFillColor)
7e argument : Position du prochain élément.
    0: à droite
    1: au début de la ligne du dessous
    2: en dessous et décalé de la longueur du bloc
8e argument : Coordonnées X
9e argument : Coordonnées Y
10e argument : Réinitialiser la hauteur de la dernière cellule
11e argument : stretch mode:
    0 : Désactivé
    1 : Mise à l'échelle horizontale uniquement si le texte est plus grand que
la largeur de la cellule
    2 : Mise à l'échelle horizontale forcée pour s'adapter à la largeur de la
cellule
    3 : Espacement des caractères uniquement si le texte est plus grand que la
largeur de la cellule
    4 : Espacement forcé des caractères pour s'adapter à la largeur de la
cellule
12e argument : Si le texte de la cellule est du HTML
13e argument : Appliquer ou non l'autopadding
14e argument : hauteur max (doit être supérieur au 1er argument)
15e argument : Alignement vertical (fonctionne que si le 11e argument
(texteHTML) = false)
    T : En haut
    M : Au milieu
    B : En bas
16e argument : ??
*/
$pdf->MultiCell(50, 15, "txtG", 1, 'L', 1, 0, '', '',
true,0,false,true,35,'T');

//Ombre sur le texte
/*
La méthode setTextShadow pend en paramètre un tableau contenant 6 éléments :
1er élément : Activer ou non les ombres
2e élément : Décalage en longueur
3e élément : Décalage en hauteur
4e élément : La couleur
```

```
5e élément : L'opacité
6e élément : Mode de fusion
*/
$pdf->setTextShadow(array('enabled'=>true, 'depth_w'=>0.2, 'depth_h'=>0.2,
'color'=>array(196,196,196), 'opacity'=>0.1, 'blend_mode'=>'Normal'));

/*
La méthode ANNOTATION prend en paramètre 7 arguments
1er argument : Position en X
2e argument : Position en Y
3e argument : Longueur du rectangle d'annotation
4e argument : Hauteur du rectangle d'annotation
5e argument : Texte d'annotation
6e argument : ???
7e argument : ???
*/
$pdf->Annotation(83, 17, 1, 10, "remarque : xxx");

//équivalent d'une balise <br> avec comme argument la hauteur
$pdf->Ln(24);

/***************
*    IMAGES    *
***************/

//Qualité du fichier JPEG à insérer
$pdf->setJPEGQuality(75);

/*
La méthode Image prend en paramètre xx arguments :
1er argument : Le nom du fichier
2e argument : Position en X
3e argument : Position en Y
4e argument : Longueur de l'imaga
5e argument : Hauteur de l'imaga
6e argument : Type de l'image (jpg, png, gif, ...)
7e argument : URL du l'image
8e argument : Indique l'alignement du pointeur à côté de l'insertion de
l'image par rapport à la hauteur de l'image
    T: top-right
    M: middle-right
    B: bottom-right
    N : Ligne suivante
9e argument :  Redimensionner l'image ou pas
10e argument : résolution en dpi
11e argument : Alignement
    L : à gauche (par défaut)
    C : centré
    R : à droite
```

```
12e argument : Vrai si l'image est un masque
13e argument : ?? (Vrai ou Faux)
14e argument : Bordure
    0 : Pas de bordure
    1 : bordure de chaque coté
    XXXX :
        T pour une bordure en haut
        R pour une bordure à droite
        B pour une bordure en bas
        L pour une bordure à Gauche
            Exemple : LR pour une bordure à gauche et  àdroite
    Array : Pour définir le type de bordure, l'épaisseur, la couleur, ...
        Exemple : array('LTRB' => array('width' => 2, 'cap' => 'butt', 'join'
=> 'miter', 'dash' => 0, 'color' => array(0, 0, 0)))
15e argument : ???
16e argument : Cacher ou non l'image
17e argument : Si vrai, l'image est redimensionnée pour ne pas dépasser les
dimensions de la page
18e argument : Attribut alt
19e argument : ???
 */
$pdf->Image('img1.jpg', '', '', 40, 40, '', 'images', 'N', false, 300, '',
false, false, 0, false, false, false);

//Ratio
$pdf->setImageScale(PDF_IMAGE_SCALE_RATIO);

/**************************
*    DESSINER DES FORMES   *
**************************/

/*
La méthode PIESECTOR prend en paramètre 7 arguments
1er argument : Position en X du centre
2e argument : Position en Y du centre
3e argument : Le rayon
4e argument : L'angle de départ
5e argument : L'angle d'arrivée
6e argument : Style du rendu
7e argument : Indique s'il faut aller dans le sens des aiguilles d'une montre
8e argument : Origine des angles (0° pour 3H, 90° pour midi, 180° pour 9H,
270° pour 6H)
*/
$pdf->PieSector($xc, $yc, $r, 0, 180, 'F', true,270);

/*
La méthode RECT prend en paramètres 7 arguments :
1er argmument : Position en X
2e argmument : Position en Y
```

```
3e argmument : La longueur
4e argmument : La hauteur
5e argmument : Le style
    F pour remplir le rectangle
    D : Pour avoir une bordure
6e argument : Le type de bordure
7e argmument : La couleur de fond
*/
$border_style = array('all' => array('width' => 2, 'cap' => 'square', 'join'
=> 'miter', 'dash' => 0, 'phase' => 0));
$pdf->Rect(65, 25, 40, 20, 'DF', $border_style,array(255,0,0));

/*
La méthode ELLIPSE prend en paramètres 11 arguments :
1er argmument : Position en X
2e argmument : Position en Y
3e argmument : L'angle en X (en radius)
4e argmument : L'angle en Y (en radius)
5e argmument : Rotation horaire ou anti-horaire
6e argmument : L'angle de début
7e argmument : L'angle de fin
8e argmument : Le style
    F pour remplir le rectangle
    D : Pour avoir une bordure
9e argument :Le type de bordure
11e argmument : ???
*/
$borderCircle = array('width' => 0.5, 'cap' => 'butt', 'join' => 'miter',
'dash' => '0', 'color' => array(0, 128, 0));
$pdf->Ellipse(175,35,15,5, 0, 0, 360, 'DF', $borderCircle,array(220, 200,
200));

/*
La méthode CIRCLE prend en paramètres 9 arguments :
1er argmument : Position en X
2e argmument : Position en Y
3e argmument : Le rayon
4e argmument : L'angle de début
5e argmument : L'angle de fin
6e argmument : Le style
    F pour remplir le rectangle
    D : Pour avoir une bordure
7e argument : Le type de bordure
8e argmument : La couleur de fond
9e argmument : ???
*/
$pdf->Circle(55, 65, 10, 0, 360, 'DF', $borderCircle, array(255,255,0));
```

```
/********************
*    LES GRADIENTS   *
********************/

/*
La méthode LINEARGRADIENT prend en paramètre 7 arguments
1er argmument : Position en X
2e argmument : Position en Y
3e argmument : La longueur
4e argmument : La hauteur
5e argument : La 1ère couleur
6e argument : La 2e couleur
7e argument : Un tableau avec les coordonnées du vecteur qui va définir le
radient. Les indexes paires correspondent aux abscisses et Les indexes
impaires correspondent aux ordonnées
*/
$pdf->LinearGradient(15, 45, 40, 40, $red, $blue, $coordsHorizontal);

/*
La méthode RADIALGRADIENT prend en paramètre 7 arguments
1er argmument : Position en X
2e argmument : Position en Y
3e argmument : La longueur
4e argmument : La hauteur
5e argument : La 1ère couleur
6e argument : La 2e couleur
7e argument : Un tableau avec les coordonnées du vecteur qui va définir le
radient. Les indexes paires correspondent aux abscisses et Les indexes
impaires correspondent aux ordonnées
*/
$pdf->RadialGradient(15, 120, 40, 40, $red, $blue, $coordsRadial1);


/*****************************
*    LES REPERES DE DECOUPE    *
*****************************/

/*
La méthode CROPMARK prend en paramètres 6 arguments :
1er argmument : Position en X
2e argmument : Position en Y
3e argmument : La longueur
4e argmument :  La hautuer
5e argmument : La postion :
    T : En haut
    L : A gauche
    B : En bas
    R : à deoite
6e argmument : La couleur
```

```
*/
$pdf->cropMark(30, 50, 10, 10, 'TL',array(125,125,125));
$pdf->cropMark(180, 50, 10, 10, 'TR',array(255,0,0));
$pdf->cropMark(30, 200, 10, 10, 'BL',array(0,255,0));
$pdf->cropMark(180, 200, 10, 10, 'BR',array(0,0,255));


/***********************
*    LES FORMULAIRES    *
***********************/

/*
La méthode TEXTFIELD prend en paramètre 8 arguments
1er argument : Le libellé
2e argument : La longueur du champ
3e argument : La hauteur du champ
4e argument : Propriété javascript
5e argument : ???
6e argument : Position en X
7e argument : Position en Y
8e argument : Autoriser ou non le javascript
*/
$pdf->TextField('firstname', 50, 5,array(),array(),'','',false);

/*
La méthode COMBOBOX permet de créer un menu déroulant et prend en paramètre 9
arguments
1er argument : Le libellé
2e argument : La longueur du champ
3e argument : La hauteur du champ
4e argument : Tableau de valeur possible
5e argument : Propriété javascript
6e argument : ???
7e argument : Position en X
8e argument : Position en Y
9e argument : Autoriser ou non le javascript
*/
$pdf->ComboBox('gender', 30, 5, array(array('', '-'), array('M', 'Homme'),
array('F', 'Femme')));

/*
La méthode LISTBOX permet de créer un menu déroulant multi-choix et prend en
paramètre 9 arguments
1er argument : Le libellé
2e argument : La longueur du champ
3e argument : La hauteur du champ
4e argument : Tableau de valeur possible
5e argument : Propriété javascript
6e argument : ???
```

```
7e argument : Position en X
8e argument : Position en Y
9e argument : Autoriser ou non le javascript
*/
$pdf->ListBox('listbox', 60, 20, array('', 'Running', 'Handball', 'Badminton',
'Tennis', 'Foot'), array('multipleSelection'=>'true'));

/*
La méthode RADIOBUTTON permet de créer des boutons radios et prend en
paramètre 9 arguments
1er argument : Le libellé
2e argument : Taille des boutons
3e argument : Propriété javascript
4e argument : ???
5e argument : Valeur
6e argument : Coché ou non par defaut
7e argument : Position en X
8e argument : Position en Y
9e argument : Autoriser ou non le javascript
*/
$pdf->RadioButton('drink', 5, array(), array(), 'Bière');

/*
La méthode CHECKBOX permet de créer des boutons radios et prend en paramètre 9
arguments
1er argument : Le libellé
2e argument : Taille des boutons
3e argument : Coché ou non par defaut
4e argument : Propriété javascript
5e argument : ???
6e argument : Valeur
7e argument : Position en X
8e argument : Position en Y
9e argument : Autoriser ou non le javascript
*/
$pdf->CheckBox('newsletter', 5, true, array(), array(), 'OK');


/*********************************
*    AFFICHAGE EN MODE JOURNAL    *
*********************************/
/*
On définit les zones de la page sans écriture pour éviter que le texte ne
chevauche les images
    'page' => page number or empy for current page
    'xt' => X top
    'yt' => Y top
    'yb' => Y bottom
    'side' => Coté de la page ('L' = left or 'R' = right)
```

```php
*/
$regions = array(
    array('page' => '', 'xt' => 153, 'yt' =>  30, 'xb' => 153, 'yb' =>  70,
'side' => 'R'),
    array('page' => '', 'xt' => 60, 'yt' =>  230, 'xb' => 60, 'yb' =>  272,
'side' => 'L'),
);
$pdf->setPageRegions($regions);


/**************************
*    TABLE DES MATIERES    *
**************************/
/*
La méthode  BOOKMARK prend en paramètre 8 arguments :
1er argument : Le nom du chapitre dans la table des matières
2e argument : Le niveau du chapitre (0 pour le 1er niveau)
3e argument : ???
4e argument : Le numéro de la page (laisser vide pour une numérotation
automatique). C'est ce paramètre qui va permettre d'accèder à la page
5e argument : Le style
    B : Gras
    I : Italique
    BI : Gras et italique
6e argument : Couleur
7e argument : ???
8e argument : ???
*/

/* la méthode SETLINK prend en paramètre 3 arguments :
1er argument : L'ID du lien retourner par la méthode AddLink()
2e argument : ???
3e argument : Le numéro de la page cible
*/
//Création du chapitre dans la table des matières
$pdf->Bookmark('Chapitre 1', 1, 0, '', 'B', array(0,64,128));
//Lien sur la page 2 qui va permette de revenir à la table des matières quand
on va cliquer dessus
$index_link = $pdf->AddLink();
$pdf->setLink($index_link, 0, '*1');

//Création de la page qui va contenir la table des matières (un peu comme la
méthode addPage)
$pdf->addTOCPage();

//Création de la table des matières
$pdf->addTOC(1, 'courier', '.', 'INDEX 2', 'B', array(255,255,0));

//Fin de la création de la table des matières
```

```php
$pdf->endTOCPage();


/************************
*    LES COORDONNÉES    *
************************/

//Obtenir la coordonnée X actuelle
$y = $pdf->getX();

//Obtenir la coordonnée Y actuelle
$y = $pdf->getY();


/*****************************
*    GÉNÉRATION DU FICHIER    *
*****************************/

/*La méthode Output prend en paramètre 2 arguments :
1er argument : le nom du fichier à sauvegarder
2e argument : le type d'enregistrement
    I : Ouvrir dans le navigateur
    D : Lancer le téléchargement
    F : Sauvegarder sur le serveur
    FD : Sauvegarder sur le serveur et Lancer le téléchargement
*/

$pdf->Output('recapFonction.pdf', 'I');


/**************************
*    À QUOI ÇA SERT ???    *
**************************/
// set default monospaced font
$pdf->setDefaultMonospacedFont(PDF_FONT_MONOSPACED);

// set some language-dependent strings (optional)
if (@file_exists(dirname(__FILE__).'/lang/eng.php')) {
    require_once(dirname(__FILE__).'/lang/eng.php');
    $pdf->setLanguageArray($l);
}

// set default font subsetting mode
$pdf->setFontSubsetting(true);
```