Opérateurs postscripts

Nota : Les commandes ci-dessous sont recopiées du *Manuel de référence du langage PostScript*, d'Adobe Systems Incorporation, chez Addison-Wesley, 1992, conformément à l'autorisation Adobe publiée à la page 11 de cet ouvrage.

1. Opérateurs de manipulation de la pile d'opérandes

any **pop** $-\longrightarrow$ enlève l'élément supérieur

any_1 any_2 **exch** any_2 $any_1 \longrightarrow$ intervertit les deux éléments en haut de la pile

any **dup** *any any* $\longrightarrow$ duplique l'élément au sommet

$any_1 \ldots any_n$ n **copy** $any_1 \ldots any_n$ $any_1 \ldots any_n \longrightarrow$ duplique n éléments au sommet

$any_n \ldots any_0$ n **index** $any_n \ldots any_0$ $any_n \longrightarrow$ duplique un élément déterminé

$a_{n-1} \ldots a_0$ n j **roll** $a_{(j-1) \bmod n} \ldots a_0$ a_{n-1} $a_{j \bmod n} \longrightarrow$ permute n éléments j fois

$\vdash$ $any_1 \ldots any_n$ **clear** $\vdash \longrightarrow$ enlève tous les éléments

$\vdash$ $any_1 \ldots any_n$ **count** $\vdash$ $any_1 \ldots any_n$ $n \longrightarrow$ compte les éléments dans la pile

$-$ **mark** *mark* $\longrightarrow$ place une marque sur la pile

marks $obj_1 \ldots obj_n$ **cleartomark** $-\longrightarrow$ enlève les éléments jusqu'à *mark*

marks $obj_1 \ldots obj_n$ **counttomark** *marks* $obj_1 \ldots obj_n$ $n \longrightarrow$ compte les élémens jusqu'à *mark*

2. Opérateurs arihmétiques et mathématiques

num_1 num_2 **add** *sum* $\longrightarrow$ num_1 plus num_2

num_1 num_2 **div** *quotient* $\longrightarrow$ num_1 divisé par num_2

int_1 int_2 **idiv** *quotient* $\longrightarrow$ division entière

int_1 int_2 **mod** *remainder* $\longrightarrow$ int_1 mod int_2

num_1 num_2 **mul** *product* $\longrightarrow$ num_1 multiplié par num_2

num_1 num_2 **sub** $difference \longrightarrow num_1$ moins num_2

num_1 **abs** $num_2 \longrightarrow$ valeur absolue de num_1

num_1 **neg** $num_2 \longrightarrow$ opposé de num_1

num_1 **ceiling** $num_2 \longrightarrow$ plafond de num_1

num_1 **floor** $num_2 \longrightarrow$ plafond de num_1

num_1 **round** $num_2 \longrightarrow$ arrondit num_1 à l'entier le plus proche

num_1 **truncate** $num_2 \longrightarrow$ enlève la partie fractionnaire de num_1

num **sqrt** *real* $\longrightarrow$ racine carrée de *num*

num den **atan** *angle* $\longrightarrow$ arc tangente de num/den en degrés

angle **cos** *real* $\longrightarrow$ cosinus de *angle* (en degrés)

angle **sin** *real* $\longrightarrow$ sinus de *angle* (en degrés)

base exponent **exp** *real* $\longrightarrow$ élève *base* à la puissance *exponent*

num **ln** *real* $\longrightarrow$ logarithme naturel (base e)

num **log** *real* $\longrightarrow$ logarithme décimal (base 10)

$-$ **rand** *int* $\longrightarrow$ génère un entier au hasard

int **srand** $-\longrightarrow$ paramètre le nombre d'origine du générateur aléatoire

$-$ **rrand** *int* $\longrightarrow$ renvoie le nombre d'origine du générateur aléatoire

3. Opérateurs de tableau

int **array** *array* $\longrightarrow$ crée un tableau de longueur *int*

$-$ **[** *mark* $\longrightarrow$ commence la construction de tableau

mark $obj_0 \ldots obj_{n-1}$ **]** *array* $\longrightarrow$ termine la construction de tableau

array **length** *int* $\longrightarrow$ nombre d'éléments dans *array*

array index **get** *any* $\longrightarrow$ obtient l'élément du tableau indexé par *index*

array index any **put** $-\longrightarrow$ met *any* dans *array* à *index*

array index count **getinterval** *subarray* $\longrightarrow$ sous-tableau de *array* commençant à *index* et long de *count* éléments

array$_1$ *index array*$_2$ **putinterval** *–* $\longrightarrow$ remplace le sous-tableau *array*$_1$ commençant à *index* par *array*$_2$

any$_0$ ... *any*$_{n-1}$ *array* **astore** *array* $\longrightarrow$ pousse l'élément depuis la pile vers array

array **aload** a_0 ... a_{n-1} *array* $\longrightarrow$ pousse tous les éléments de *array* dans la pile

array$_1$ *array*$_2$ **copy** *subarray*$_2$ $\longrightarrow$ copie les éléments de *array*$_1$ dans le sous-tableau initial de *array*$_2$

array proc **forall** *–* $\longrightarrow$ éxécute *proc* pour chaque élément dans *array*

4. Opérateurs de dictionnaire

int **dict** *dict* $\longrightarrow$ crée un dictionnaire ayant une contenance de *int* éléments

– **<<** *mark* $\longrightarrow$ commence la construction de dictionnaire

mark key$_1$ *value*$_1$ ... *key*$_n$ *value*$_n$ **>>** *dict* $\longrightarrow$ termine la construction de dictionnaire

dict **length** *int* $\longrightarrow$ nombre de couples clé-valeur dans *dict*

dict **maxlength** *int* $\longrightarrow$ capacité courante de *dict*

dict **begin** *–* $\longrightarrow$ pousse *dict* dans la pile des dictionnaires

– **end** *–* $\longrightarrow$ expulse la pile des dictionnaires

key value **def** *–* $\longrightarrow$ associe *key* et *value* dans le dictionnaire courant

key **load** *value* $\longrightarrow$ recherche *key* dans la pile des dictionnaires et renvoie le *vlue* qui y est associé

key value **store** *–* $\longrightarrow$ remplace la définition la plus haute de *key*

dict key value **put** *–* $\longrightarrow$ associe *key* à *value* dans *dict*

dict key **undef** *–* $\longrightarrow$ enlève *key* et *value* dans *dict*

dict key **known** *bool* $\longrightarrow$ test si *key* est dans *dict*

key **where** *dict true* ou bien *false* $\longrightarrow$ trouve le dictionnaire dans lequel *key* est défini

dict proc **forall** *–* $\longrightarrow$ exécute *proc* pour chaque élément de *dict*

– **currentdict** *dict* $\longrightarrow$ pousse le dictionnaire courant dans la pile des opérandes

– **errordict** *dict* $\longrightarrow$ dictionnaire des gestions d'erreur

– **$error** *dict* $\longrightarrow$ dictionnaire de contrôle des erreurs et de statut

– **systemdict** *dict* $\longrightarrow$ dictionnaire système

– **userdict** *dict* $\longrightarrow$ dictionnaire en écriture en VM locale

– **globaldict** *dict* $\longrightarrow$ dictionnaire en écriture en VM globale

– **statusdict** *dict* $\longrightarrow$ dictionnaire dépendant du produit

– **countdictstack** *int* $\longrightarrow$ compte les éléments dans la pile des dictionnaires

array **dictstack** *subarray* $\longrightarrow$ copie la pile des dictionnaires dans *array*

– **cleardictstack** *–* $\longrightarrow$ enlève tous les dictionnaires non permanents de la pile des dictionnaires

5. Opérateurs de chaînes

int **string** *string* $\longrightarrow$ crée une chaîne de longueur *int*

string **length** *int* $\longrightarrow$ nombre d'éléments dans *string*

string index **get** *int* $\longrightarrow$ obtinet l'élement de *string* indexé par *index*

string index int **put** *–* $\longrightarrow$ place *int* dans *string* à *index*

string index count **getinterval** *substring* $\longrightarrow$ sous-chaîne de *string* commençant à *index* pour *count* éléments

string$_1$ *index string*$_2$ **putinterval** *–* $\longrightarrow$ remplace la sous-chaîne de *string*$_1$ commençant à *index* par *string*$_2$

string$_1$ *string*$_2$ **copy** *substring*$_2$ $\longrightarrow$ copie les élément de *string*$_1$ dans la sousè-chaîne initiale de *string*$_2$

string proc **forall** *–* $\longrightarrow$ exécute *proc* pour chaque élément de *string*

string seek **anchorsearch** *post match true* ou bien *string false* $\longrightarrow$ détermine si *seek* est une sous-chaîne de *string*

string seek **search** *post match pre true* ou bien *string false* $\longrightarrow$ cherche *seek* dans *string*

string **token** *post token true* ou bien *string false* $\longrightarrow$ lit le léxème à partir du début de *string*

6. Opérateurs relationnels, booléens et bit à bit

any$_1$ *any*$_2$ **eq** *bool* ⟶ test l'égalité

any$_1$ *any*$_2$ **ne** *bool* ⟶ test l'inégalité

num$_1$/*str*$_1$ *num*$_2$/*str*$_2$ **ge** *bool* ⟶ teste si supérieur ou égal

num$_1$/*str*$_1$ *num*$_2$/*str*$_2$ **gt** *bool* ⟶ teste si supérieur

num$_1$/*str*$_1$ *num*$_2$/*str*$_2$ **le** *bool* ⟶ test si inférieur

num$_1$/*str*$_1$ *num*$_2$/*str*$_2$ **lt** *bool* ⟶ teste si inférieur ou égal

bool$_1$/*int*$_1$ *bool*$_2$/*int*$_2$ **and** *bool*$_3$/*int*$_3$ ⟶ et logique bit à bit

bool$_1$/*int*$_1$ **not** *bool*$_2$/*int*$_2$ ⟶ non logique bit à bit

num$_1$/*string*$_1$ *num*$_2$/*string*$_2$ **or** *bool*$_3$/*int*$_3$ ⟶ ou inclusif logique bit à bit

num$_1$/*string*$_1$ *num*$_2$/*string*$_2$ **xor** *bool*$_3$/*int*$_3$ ⟶ ou exclusif logique bit à b it

– **true** *true* ⟶ pousse la valeur booléenne *true* (vrai)

– **false** *false* ⟶ pousse la valeur booléenne *false* (faux)

int$_1$ *shift* **bitshift** *int*$_2$ ⟶ décalement bit à bit de *int*$_1$ (positif à gauche)

7. Opérateurs de contrôle

any **exec** – ⟶ exécute un objet arbitraire

bool proc **if** – ⟶ exécute *proc* si *bool* est *true*

bool proc$_1$ *proc*$_2$ **ifelse** – ⟶ exécute *proc*$_1$ si *bool* est *true*, *proc*$_2$ sinon

init incr limit proc **for** – ⟶ exécute *proc* avec les valeurs à partie de *init* par pas de *inc* jusqu'à *limit*

int proc **repeat** – ⟶ exécute *int* fois *proc*

proc **loop** – ⟶ exécute *proc* un nombre indéfini de fois

– **exit** – ⟶ quitte la boucle active la plus interne

– **stop** – ⟶ termine le contexte **stopped**

any **stopped** *bool* ⟶ établit le contexte pour attraper **stop**

– **countexecstack** *int* ⟶ compte les éléments dans la pile d'éxécution

array **execstack** *subaray* ⟶ copie la pile d'exécution dans *array*

– **quit** – ⟶ quitte l'interpréteur

– **start** – ⟶ exécuté au lancement de l'interpréteur

8. Opérateurs de type, attribut et conversion

any **type** *name* ⟶ renvoie le nom identifiant le type de *any*

any **cvlit** *any* ⟶ convertit l'objet en littéral

any **cvx** *any* ⟶ convertit l'objet en exécutable

any **xcheck** *bool* ⟶ teste l'attribut d'exécutable

array/*packedarray*/*file*/*string* **executeonly** – *array*/*packedarray*/*file*/*string* ⟶ réduit l'accès à exécutablea seulement

array/*packedarray*/*dict*/*file*/*string* **noaccess** – *array*/*packedarray*/*dict*/*file*/*string* ⟶ interdit tout accès

array/*packedarray*/*dict*/*file*/*string* **readonly** – *array*/*packedarray*/*dict*/*file*/*string* ⟶ réduit l'accès à lecture seule

array/*packedarray*/*dict*/*file*/*string* **rcheck** *bool* ⟶ teste l'accès en lecture

array/*packedarray*/*dict*/*file*/*string* **wcheck** *bool* ⟶ teste l'accès en écriture

num/*string* **cvi** *int* ⟶ convertit en entier

string **cvn** *name* ⟶ convertit en nom

num/*string* **cvr** *real* ⟶ convertit en réel

num radix string **cvrs** *substring* ⟶ convertit en chaîne avec racine

any string **cvs** *substring* ⟶ convertit en chaîne

9. Opérateurs de fichier

string$_1$ *string*$_2$ **file** *file* ⟶ ouvre le fichier idéntifié par *string*$_1$ avec l'accès *string*$_2$

src/*tgt*/*param*$_1$ ... *param*$_n$ *name* **filter** *file* ⟶ établit le fichier filtré

file **closefile** – ⟶ ferme *file*

file **read** *int true* ou bien *false* ⟶ lit un caractère dans *file*

file int **write** – ⟶ écrit un caractère dans *file*

file string **readhexstring** *substring bool* ⟶ lit une chaîne hexadécimale depuis *file* vers *string*

file string **writehexstring** – ⟶ écrit *string* dans *file* en chaîne hexadécimale

file string **readstring** *substring bool* ⟶ lit une ligne depuis *file* dans *string*

file string **writestring** – ⟶ écrit *string* dans *file*

file string **readline** *substring bool* ⟶ lit une ligne depuis *file* dans *string*

file **token** *token true* ou bien *false* ⟶ lit un léxème depuis *file*

file **bytesavailable** *int* ⟶ nombre d'octets pouvant être lus

– **flush** – ⟶ renvoie les données du tampon dans le fichier de sortie standard

file **flushfile** – ⟶ renvoie les données du tampon ou les lit jusqu'à EOF

file **resetfile** – ⟶ ignore les caractères dans le tampon

file **status** *bool* ⟶ renvoie le statut de *file*

string **status** *pages bytes referenced created true* ou bien *false* ⟶ renvoie l'information à propos du fichier nommé

string **run** – ⟶ exécute le contenu du fichier nommé

– **currentfile** *file* ⟶ renvoie le fichier en cours d'exécution

any$_1$ ... *any*$_n$ *string* **deletefile** – ⟶ détruit le fichier nommé

string$_1$ *string*$_2$ **renamefile** – ⟶ renomme le fichier appelé *string*$_1$ en *string*$_2$

template proc scratch **filenameforall** – ⟶ exécute *proc* pour chaque nom de fichier correspondant à *template*

file int **setfileposition** – ⟶ met *file* à la position indiquée

file **fileposition** *int* ⟶ renvoie la position courant dans *file*

string **print** – ⟶ écrit *string* dans le fichier de sortie standard

any **=** – ⟶ écrit la représentation textuelle de *any* dans le fichier de sortie standard

any **==** – ⟶ écrit la représentation syntaxique de *any* dans le fichier de sortie standard

⊢ *any*$_1$ ... *any*$_n$ **stack** – ⊢ *any*$_1$ ... *any*$_n$ ⟶ imprime la pile de façon non destructive en utilisant **=**

⊢ *any*$_1$ ... *any*$_n$ **pstack** – ⊢ *any*$_1$ ... *any*$_n$ ⟶ imprime la pile de façon non destructive en utilisant **==**

obj int **printobject** – ⟶ écrit un objet binaire dans le fichier de sortie standard, en utilisant *int* en tant que drapeau

file obj int **writeobject** – ⟶ écrit un objet binaire dans *file* en utilisant *int* en tant que drapeau

int **setobjectformat** – ⟶ définit le format d'objet binaire (0 = désactivé, 1 = IEEE haut, 2 = bas, 3 = haut natif, 4 = bas)

– **currentobjectformat** *int* ⟶ renvoie le format d'objet binaire

10. Opérateurs de ressource

key instance category **defineresource** *instance* ⟶ enregistre la ressource nommée *instance* dans *category*

key category **undefinedresource** – ⟶ enlève l'enregistrement de ressource

key category **findresource** *instance* ⟶ renvoie la ressource *instance* identifiée par *key* dans *category*

key category **resourcestatus** *status size true* ou bien *false* ⟶ renvoie le *status* des instances de ressource

template proc scratch category **resourceforall** – ⟶ énumère les instances de ressources dans *category*

11. Opérateurs de mémoire virtuelle

– **save** *save* ⟶ crée une photo de la VM

save **restore** – ⟶ réinstalle la photo de la VM

bool **setglobal** – ⟶ définit le mode d'allocation en VM (*fale* = local, *true* = global)

– **currentglobal** *bool* ⟶ renvoie le mode courant d'allocation en VM

any **gcheck** *bool* ⟶ *true* si *any* est simple ou en VM globale, *false* si en VM locale

$bool_1$ *password* **startjob** $bool_2$ ⟶ commence une nouvelle tâche qui modifiera la VM initiale si *bool* est vrai

index any **defineuserobject** – ⟶ définit l'objet utilisateur associé à *index*

index **executeuserobject** – ⟶ exécute l'objet utilisateur associé à *index*

index **undefineduserobject** – ⟶ enlève l'objet utilisateur associé à *index*

– **UserObject** *array* ⟶ tableau **UserObjects** courant défini dans **userdict**

12. Opérateurs divers

proc **bind** *proc* ⟶ remplace les noms d'opérateurs dans *proc* par les opérateurs

– **null** *null* ⟶ pousse *null* dans la pile d'opérandes

– **version** *string* ⟶ version de l'interpréteur

– **realtime** *int* ⟶ renvoie le temps réel en millisecondes

– **usertime** *int* ⟶ renvoie le temps d'exécution en millisecondes

– **languagelevel** *int* ⟶ niveau de fonction du langage

– **product** *string* ⟶ nom du produit

– **revision** *int* ⟶ numéro de révision du produit

– **serialnumber** *int* ⟶ numéro de série de la machine

– **executive** – ⟶ appelle l'exécuteur intéractif

bool **echo** – ⟶ alterne l'écho actif et inactif

– **prompt** – ⟶ exécuté quand prêt pour l'entrée intéractive

13. Opérateurs de de l'état graphique - Indépendants du périphérique

– **save** – ⟶ pousse l'état graphique

– **grestore** – ⟶ remet l'état graphique en place

– **grestorealll** – ⟶ remet l'état graphique le plus ancien en place

– **initgraphics** – ⟶ réinitialise les paramètres de l'état graphique

– **gstate** *gstate* ⟶ crée un objet état graphique

gstate **setgstate** – ⟶ établit l'état graphique à partir de *gstate*

gstate **currentgstate** *gstate* ⟶ copie l'état graphique courant dans *gstate*

num **setlinewidth** – ⟶ ajuste la largeur de ligne

– **currentlinewidth** *num* ⟶ renvoie la largeur de ligne courante

int **setlinecap** – ⟶ définit la forme des fins de ligne lors des tracés (0 = en ogive, 1 = rond, 2 = en biais)

– **currentlinecap** *int* ⟶ renvoie le recouvrement courant de la ligne

int **setlinejoin** – ⟶ définit la forme des coins lors du tracé (0 = en ogive, 1 = rond, 2 = en biais)

– **currentlinejoin** *num* ⟶ renvoie le type de jointure courant des lignes

num **setmiterlimit** – ⟶ définit la limite de longueur d'intersection

– **currentmiterlimit** *num* ⟶ renvoie la valeur courante du paramètre de limite d'extrémité de ligne

bool **setstrokeadjust** – ⟶ définit l'ajustement du tracé (*false* = désactivé, *true* = activé)

– **currentstrokeadjust** *bool* ⟶ renvoie l'ajustement de tracé courant

array offset **setdash** – ⟶ définit le pointillé pour le tracé

– **currentdash** *array offset* ⟶ renvoie le pointillé de tracé courant

array **setcolorspace** – ⟶ définit l'espace de couleur

– **currentcolorspace** *array* ⟶ renvoie l'espace de couleur courant

$comp_1 \ldots comp_n$ **setcolor** – ⟶ définit les composantes de couleur

– **currentcolor** $comp_1 \ldots comp_n$ ⟶ renvoie les composantes courantes de couleur

num **setgray** – ⟶ définit l'espace de couleur pour **DeviceGray** et la couleur de la valeur de gris spécifiée (0 = noir, 1 = blanc)

– **currentgray** *num* ⟶ renvoie la valeur courante en tant que valeur de gris

hue sat brt **sethsbcolor** – ⟶ définit **DeviceRGB** comme espace de couleur et la couleur à la teinte, saturation et luminosité indiquée

– **currenthsbcolor** *hue sat brt* ⟶ retourne la couleur courant en teinte, saturation et luminosité

red green blue **setrgbcolor** – ⟶ définit **DeviceRGB** comme espace de couleur et la couleur aux rouge, vert et bleu indiqués

– **currentrgbcolor** *red green blue* ⟶ renvoie la couleur courante selon ses composantes rouge, vert, bleu

cyan magenta yellow black **setcmykcolor** – ⟶ définit **DeviceCYMK** comme espace de couleur et la couleur selon le cyan, jaune, magenta et noir indiqués

– **currentcmykcolor** *cyan magenta yellow black* ⟶ renvoie la couleur courane selon les cyan, jaune, magenta et noir

14. Opérateurs de de l'état graphique - Dépendants du périphérique

dict **sethalftone** – ⟶ définit le dictionnaire de simili

– **currenthalftone** *dict* ⟶ renvoie le dictionnaire de simili courant

frequency angle proc **setscreen** – ⟶ définit l'écran de simili de gris

– **currentscreen** *frequency angle proc* ⟶ renvoie l'écran de simili de gris courant

redfreq redang redproc greenfreq greenang greenproc bluefreq blueang blueproc grayfreq grayang grayproc **setcolorscreen** – ⟶ définit les quatres écrans de simili

– **currentcolorscreen** *redfreq redang redproc greenfreq greenang greenproc bluefreq blueang blueproc grayfreq grayang grayproc* ⟶ renvoie les quatres écrans de simili

proc **settransfer** – ⟶ définit la fonction de transfert de gris

– **currenttransfer** *proc* ⟶ renvoie la fonction de transfert de gris

redproc greenproc blueproc grayproc **setcolortransfer** – ⟶ définit les quatre fonctions de transfert

– **currentcolortransfer** *redproc greenproc blueproc grayproc* ⟶ renvoie les fonctions de transfert courantes

proc **setblackgeneration** – ⟶ définit la fonction de génération de noir courante

– **currentblackgeneration** *proc* ⟶ renvoie la fonction de génération de noir courante

proc **setundercolorremoval** – ⟶ définit la fonction d'enlèvement de couleur sous-jacente

– **currentundercolorremoval** *proc* ⟶ renvoie la fonction d'enlèvement de couleur sous-jacente courante

dict **setcolorrenderring** – ⟶ définit le dictionnaire de rendu des couleurs basées sur le CIE

– **currentcolorrenderring** *dict* ⟶ renvoie le dictionnaire courant de rendu des couleurs basées sur le CIE

num **setflat** – ⟶ définit la tolérance de l'arrondi des courbes

– **currentflat** *num* ⟶ renvoie la tolérance de l'arrondi des courbes

bool **setoverprint** – ⟶ définit le paramètre de sur-impression

– **currentoverprint** *bool* ⟶ renvoie le paramètre de sur-impression courant

15. Opérateurs de système de coordonnées et de matrice

– **matrix** *matrix* ⟶ crée une matrice identité

– **initmatrix** – ⟶ définit la CTM pour le périphérique par défaut

matrix **identmatrix** *matrix* ⟶ remplit *matrix* avec la transformation identité

matrix **defaultmatrix** *matrix* ⟶ remplit *matrix* avec la matrice par défaut du périphérique

matrix **currentmatrix** *matrix* ⟶ remplit *matrix* avec la CTM

matrix **setmatrix** – ⟶ remplace la CTM ar *matrix*

t_x t_y **translate** – ⟶ déplace l'espace utilisateur par (t_x, t_y)

t_x t_y *matrix* **translate** *matrix* ⟶ définit le déplacement par (t_x, t_y)

S_x S_y **scale** – ⟶ met à l'échelle l'espace utilisateur par s_x et s_y

S_x S_y *matrix* **scale** *matrix* ⟶ définit la mise à l'échelle par s_x et s_y

angle **rotate** – ⟶ effectue une rotation de l'espace utilisateur de *angle* degrés

angle matrix **rotate** *matrix* ⟶ définit la rotation par *angle* degrés
matrix **concat** – ⟶ remplace la CTM par le produit *matrix* × CTM
$matrix_1$ $matrix_2$ $matrix_3$ **concatmatrix** $matrix_3$ ⟶ remplit $matrix_3$ avec le produit $matrix_1 \times matrix_2$
x y **transform** x' y' ⟶ transforme (x, y)*parCTM*
x y matrix **transform** x' y' ⟶ transform (x, y) par *matrix*
dx dy **dtransform** dx' dy' ⟶ transforme la distance (dx, dy) par CTM
dx dy matrix **dtransform** dx' dy' ⟶ transforme la distance (dx, dy) par *matrix*
x' y' **itransform** *x y* ⟶ inverse la transformation (x', y') par CTM
x' y' *matrix* **itransform** *x y* ⟶ inverse la transformation (x', y') par *matrix*
dx' dy' **idtransform** *dx dy* ⟶ inverse la transformation de la distance (dx', dy') par CTM
dx' dy' *matrix* **idtransform** *dx dy* ⟶ inverse la transformation de la distance (dx', dy') par *matrix*
$matrix_1$ $matrix_2$ **invertmatrix** $matrix_2$ ⟶ remplit $matrix_2$ avec l'inverse de $matrix_1$

16. Opérateurs de construction de chemin

– **newpath** – ⟶ initialise et vide le chemin courant
– **currentpoint** *x y* ⟶ renvoie les coordonnées du point courant
x y **moveto** – ⟶ définit le point courant à (x, y)
dx dy **rmoveto** – ⟶ **moveto** relatif
x y **lineto** – ⟶ ajoute une ligne droite jusqu'en (x, y)
dx dy **rlineto** – ⟶ **lineto** relatif
x y r ang_1 ang_2 **arc** – ⟶ ajoute un arc dans le sens contraire des aiguilles d'une montre
x y r ang_1 ang_2 **arcn** – ⟶ ajoute un arc dans le sens des aiguilles d'une montre
x_1 y_1 x_2 y_2 *r* **arct** *r* ⟶ ajoute un arc tangent
x_1 y_1 x_2 y_2 *r* **arcto** xt_1 yt_1 xt_2 yt_2 ⟶ ajoute un arc tangent
x_1 y_1 x_2 y_2 x_3 y_3 **curveto** – ⟶ ajoute une section cubique de Bézier
dx_1 dy_1 dx_2 dy_2 dx_3 dy_3 **rcurveto** – ⟶ **curveto** relatif
– **closepath** – ⟶ connecte le sous-chemin à son point de départ
– **flattenpath** – ⟶ convertit les courbes en suites de segments de droites
– **reversepath** – ⟶ renverse la direction du chemin courant
– **strokepath** – ⟶ calcul le contour du chemin tracé
userpath **ustrokepath** – ⟶ calcul le contour du *userpath* tracé
userpath matrix **ustrokepath** – ⟶ calcul le contour du *userpath* tracé
string bool **charpath** – ⟶ ajoute un contour de caractère au chemin courant
userpath **uappend** – ⟶ interprète *userpath* et l'ajoute au chemin courant
– **clippath** – ⟶ définit le chemin d'incrustation en tant que chemin courant
ll_x ll_y ur_x ur_y **setbbox** – ⟶ définit le cadre de limite pour le chemin courant
– **pathbbox** ll_x ll_y ur_x ur_y ⟶ renvoie le cadre de limite pour le chemin courant
move line curve close **pathforall** – ⟶ détaille le chemion courant
bool **upath** *userpath* ⟶ crée *userpath* pour le chemin courant ; inclut **ucache** si *bool* est *true*
– **initclip** – ⟶ définit le chemin d'incrustation au périphérique par défaut
– **clip** – ⟶ incruste en utilisant la règle du nombre sinueux différent de zéro
– **eoclip** – ⟶ incruste en utilisant la règle du pair-impair
x y width height **rectclip** – ⟶ incruste avec un chemin rectangulaire
numarray/*numstring* **rectclip** – ⟶ incruste avec des chemins rectangulaires
– **ucache** – ⟶ déclare que le chemin utilisateur doît être mis en cache

17. Opérateurs de dessin

– **erasepage** – ⟶ dessine la page courante en blanc
– **fill** – ⟶ remplit le chemin courant avec la couleur courante
– **eofill** – ⟶ remplit en utilisant la règle pair-impair

– **stroke** – ⟶ dessine la ligne le long du chemin courant

userpath **ufill** – ⟶ interprète et remplit *userpath*

userpath **ueofill** – ⟶ remplit *userpath* en utilisant la règle pair-impir

userpath **ustroke** – ⟶ interprète et trace *userpath*

userpath matrix **ustroke** – ⟶ interprète *userpath*, concatène *matrix* et trace

x y width height **rectfill** – ⟶ remplit le chemin rectangulaire

numarray/*numstring* **rectfill** – ⟶ remplit les chemins rectangulaires

x y width height **rectstroke** – ⟶ trace le chemin rectangulaire

numarray/*numstring* **rectstroke** – ⟶ trace les chemins rectangulaires

dict **image** – ⟶ dessine une image numérisée

widthheightbits/*sampmatrixdatasrc* **image** – ⟶ dessine une image numérisée monochrome

18. Opérateurs de test de position à l'intérieur du chemin

x y **infill** *bool* ⟶ teste si le point (x, y) est dessiné par **fill**

userpath **infill** *bool* ⟶ teste si les pixels dans *userpath* sont dessinés par **fill**

x y **ineofill** *bool* ⟶ teste si le point (x, y) est dessiné par **eofill**

userpath **ineofill** *bool* ⟶ teste si les pixels dans *userpath* sont dessinés par **eofill**

x y userpath **inufill** *bool* ⟶ teste si le point (x, y) est dessiné par **ufill** de *userpath*

$userpath_1$ $userpath_2$ **inufill** *bool* ⟶ teste si les pixels dans $userpath_1$ sont dessinés par **ufill** dans $userpath_2$

x y userpath **inueofill** *bool* ⟶ teste si le point (x, y) est dessiné par **eofill** de *userpath*

$userpath_1$ $userpath_2$ **inueofill** *bool* ⟶ teste si les pixels dans $userpath_1$ sont dessinés par **ueofill** dans $userpath_2$

x y **instroke** *bool* ⟶ teste si le point (x, y) est dessiné par **stroke**

x y userpath **instroke** *bool* ⟶ teste si le point (x, y) est dessiné par **ustroke** de *userpath*

x y userpath matrix **inustroke** *bool* ⟶ teste si le point (x, y) est dessiné par **ustroke** de *userpath*

$userpath_1$ $userpath_2$ **inustroke** *bool* ⟶ teste si les pixels dans $userpath_1$ sont dessinés par **ustroke** de $userpath_2$

$userpath_1$ $userpath_2$ *matrix* **inustroke** *bool* ⟶ teste si les pixels dans $userpath_1$ sont dessinés par **ustroke** de $userpath_2$

19. Opérateurs de formes et de motifs

pattern matrix **makepattern** *pattern* ⟶ crée une instance d'un motif à partir d'un prototype

$comp_1 \ldots comp_n$ *pattern* **setpattern** – ⟶ installe *pattern* en tant que couleur courante

$form$ **execform** – ⟶ dessin $form$

20. Opérateurs de configuration et de sortie du périphérique

– **showpage** – ⟶ transmet et réinitialise la page courante

– **copypage** – ⟶ transmet la page courante

dict **setpagedevice** – ⟶ installe le périphérique de sortie orienté page

– **currentpagedevice** *dict* ⟶ renvoie les paramètres du périphérique courant de sortie de page

– **nulldevice** – ⟶ installe un périphérique sans sortie

21. Opérateurs de caractères et de polices

key $font$ **definefont** $font$ ⟶ enregistre $font$ comme dictionnaire de police

key **undefinedfont** – ⟶ supprime l'identification de police

key **findfont** $font$ ⟶ renvoie le dictionnaire de police identifié par *key*

$font$ *scale* **scalefont** $font'$ ⟶ met à l'échelle $font$ par *sclae* pour produire la nouvelle $font'$

$font$ *matrix* **makefont** $font'$ ⟶ transforme $font$ par *matrix* pour produire la nouvelle $font'$

$font$ **setfont** – ⟶ définit le dictionnaire de police dans l'état graphique

– **currentfont** $font$ ⟶ renvoie le dictionnaire de police courant

– **rootfont** $font$ ⟶ renvoie le dictionnaire racine d'une police composite

key scale/*matrix* **selectfont** – ⟶ définit le dictionnaire de police par son nom et le transforme

string **show** – ⟶ déssine les caractères de *string* sur la page

a_x a_y *string* **ashow** – ⟶ ajoute (a_x, a_y) à la largeur de tout caractère tout en montrant *string*

c_x c_y *char string* **widthshow** – ⟶ ajoute (c_x, c_y) à la largeur de *char* tout en montrant *string*

c_x c_y *char* a_x a_y *string* **awidthshow** – ⟶ combine les effets de **ashow** et de **widthshow**

string numarray/*numstring* **xshow** – ⟶ dessine les caractères de *string* en utilisant les largeurs *x* dans *numarray*/*numstring*

string numarray/*numstring* **xyshow** – ⟶ dessine les caractères de *string* en utilisant les largeurs *x* et *y* dans *numarray*/*numstring*

string numarray/*numstring* **yshow** – ⟶ dessine les caractères de *string* en utilisant les largeurs *y* dans *numarray*/*numstring*

name **glypshow** – ⟶ dessine les caractères identifiés par *name*

string **stringwidth** w_x w_y ⟶ largeur de *string* dans la police courante

proc string **cshow** – ⟶ appelle l'algorithme d'affichage et appelle *proc*

proc string **kshow** – ⟶ exécute *proc* entre les caractères montrés depuis *string*

– **FontDirectory** *dict* ⟶ dictionnaire des dictionnaires de police

– **GlobalFontDirectory** *dict* ⟶ dictionnaire des dictionnaires de police dans la VM globale

– **StandardEncoding** *array* ⟶ vecteur d'encodage standard des polices Adobe

– **ISOLatin1Encoding** *array* ⟶ vecteur d'encodage international ISO Latin-1 des polices

key **findencoding** *array* ⟶ trouve le tableau d'encodage

w_x w_y ll_x ll_y ur_x ur_y **setcachedevice** – ⟶ déclare les mesures des caractères en cache

$w0_x$ $w0_y$ ll_x ll_y ur_x ur_y $w1_x$ $w1_y$ ll_x ll_y v_x v_y **setcachedevice2** – ⟶ déclare les mesures des caractères en cache

w_x w_y **setcharwidth** – ⟶ déclare les mesures des caractères non mis en cache

22. Opérateurs de paramétrage de l'interpréteur

dict **setsystemparams** – ⟶ définit les paramètres de système pour l'interpréteur

– **currentsystemparams** *dict* ⟶ renvoie les paramètres de système pour l'interpréteur

dict **setusersparams** – ⟶ établit les paramètres de l'interpréteur par contexte

– **currentusersparams** *dict* ⟶ renvoie les paramètres de l'interpréteur par contexte

string dict **setdevparams** – ⟶ définit les paramètres pour le périphérique d'entrée-sortie

string **currentdevarams** *dict* ⟶ renvoie les paramètres du périphérique

int **vmreclaim** – ⟶ contrôle le ramassage des poubelles

int **setvmhreshold** – ⟶ contrôle le ramassage des poubelles

– **vmstatus** *level used maximum* ⟶ rapport sur le statut de la VM

– **cachestatus** *bsize bmax msize mmax csize cmax blimit* ⟶ renvoie le statut du cache de police et ses paramètres

num **setcachelimit** – ⟶ définit le nombre maximum d'octets pour les caractères mis en cache

mark size lower upper **setcacheparams** – ⟶ modofie les paramètres du cache de police

– **currentcacheparams** *mark size lower upper* ⟶ renvoie les paramètres courant du cache de police

mark blimit **setucacheparams** – ⟶ définit les paramètres du cache de chemin utilisateur

– **ucachestatus** *mark bsize bmax rsize rmax blimit* ⟶ renvoie le statut du cache de chemin utilisateur et les paramètres

23. Opérateurs Display PostScript

– **currentcontext** *context* ⟶ renvoie l'identificateur du contexte courant

mark obj_1 ... obj_n *proc* **fork** *context* ⟶ crée un contexte exécutant *proc* avec obj_1 ... obj_n comme opérandes

context **join** *mark* obj_1 ... obj_n ⟶ attend la fin d'un contexte et renvoie son résultat

context **detach** – ⟶ permet à un contexte de se terminer immédiatement lorsqu'il est fini

– **lock** *lock* ⟶ crée un objet verrou

lock proc **monitor** – ⟶ exécute *proc* tout en gardant *lock*

– **condition** *condition* ⟶ crée un objet condition

lock condition **wait** – ⟶ relâche *lock*, attend *condition*, reprend *lock*

condition **notify** – ⟶ reprend le contexte en attendant *condition*

– **yield** – ⟶ met momentanément en attente dans le contexte courant

index name **defineusername** – ⟶ définit un index de nom encodé

– **viewclip** – ⟶ définit la vue d'incrustation depuis le chemin courant

– **eoviewclip** – ⟶ définit la vue d'incrustation en utilisant la règle pair-impair

x y width height **rectviewclip** – ⟶ définit un chemin rectangulaire de vue d'incrustation

numarray/numstring **rectviewclip** – ⟶ définit des chemins rectangulaires de vue d'incrustation

– **initviewclip** – ⟶ réinitialise la vue d'incrustation

– **viewclippath** – ⟶ définit le chemin courant à partir de la vue d'incrustation

– **deviceinfo** *dict* ⟶ renvoie le dictionnaire contenant les informations à propos du périphérique courant

– **wtranslation** *x y* ⟶ renvoie la traduction à partir de l'origine de la fenêtre vers l'origine de l'espace du périphérique

x y **sethalftonephase** – ⟶ définit la phase de simili

– **currenthalftonephase** *x y* ⟶ renvoie la phase de simili courante

24. Erreurs

configurationerror ⟶ une demande **setpagedevice** ne peut être satisfaite

dictfull ⟶ plus de place dans le dictionnaire

dictstackoverflow ⟶ trop de **begin**

dictstackunderflow ⟶ trop de **end**

execstackoverflow ⟶ imbrication trop nombreuse d'**exec**

handleerror ⟶ appelé pour donner un rapport sur les erreurs

interrupt ⟶ demande extérieure d'interruption

invalidaccess ⟶ essai de violation des attributs d'accès

invalidcontext ⟶ utilisation impropre de l'opération concernant le contexte

invalidexit ⟶ **exit** n'est pas dans la boucle

invalidfileaccess ⟶ chaîne d'accès onn correcte

invalidfont ⟶ nom de police ou de dictionnaire incorrect

invalid ⟶ identificateur invalide pour un objet externe

invalidrestore ⟶ **restore** incorrect

ioerror ⟶ erreur d'entrée/sortie

limitcheck ⟶ dépassement des limites de l'implémentation

nocurrentpoint ⟶ le point courant n'est pas défini

rangecheck ⟶ opérande en dehors des limites

stackoverflow ⟶ dépassement de capacité de la pile d'opérandes

stackunderflow ⟶ pas assez d'opérandes dans la pile

syntaxerror ⟶ erreur de syntaxe en PostScript

timeout ⟶ dépassement de la limite de temps

typecheck ⟶ opérande de mauvais type

undefined ⟶ nom inconnu

undefinedfilename ⟶ fichier non trouvé

undefinedsource ⟶ instance de resssource non trouvée

undefinedresult ⟶ résultat insuffisant, dépassant ou sans signification

undefinedmark ⟶ marque attendue absente de la pile

unregistered ⟶ erreur interne

VMerror ⟶ VM épuisée