

30/08/02

IMPORTANT : il ne s'agit pas de la version finale de l'article. Merci d'excuser les fautes de français que vous pourriez trouver. L'article final du #9 pourra être un peu différent (mais les résultats, eux, ne bougeront pas ! ;-))

P4D Challenge #2 :

[coupé] ;-)

Challenge précédent : OrganizeRectangles

Le challenge de *Planète 4D* N°9 était le premier challenge commun aux deux éditions, *Planète 4D* et *Planet 4D*. Le nombre de développeurs inscrits à la mini-mailing list du challenge a connu une progression spectaculaire, exponentielle à cette occasion. Reportez-vous à l'encadré pour des informations sur cette liste.

Il s'agissait de ranger des rectangles de surfaces différentes (appelés *petits rectangles*) dans un rectangle plus grand, appelé le *terrain*. Les challengers devaient faire tenir le plus de *petits rectangles* possibles sur le *terrain*. Et cela n'est pas chose facile, manifestement : seuls 4 abonnés ont participé à ce challenge. Il semble que la résolution de ce *calepinage* soit un problème non pas insoluble, mais surtout trop long à résoudre.

En voyant les noms des participants, les lecteurs de l'édition française de *Planète 4D* ne seront pas étonnés : il s'agit de plusieurs piliers de la saison précédente. Gageons que les abonnés de l'édition anglaise ont simplement été un peu intimidés !

Revenons à nos moutons. Les challengers devaient écrire une méthode dont la syntaxe était la suivante :

```
OrganizeRectangles(->rectsÀRanger;terrainGauche; terrainHaut; terrainDroit; terrainBas; ->rectsDansTerrain;->rectsNonRangés)
```

Les paramètres sont détaillés dans l'énoncé (cf *Planète 4D* N°8), voici un bref rappel de leurs significations :

` Vers un BLOB contenant des informations sur les *petits rectangles*.

C_POINTEUR(\$1)

` Coordonnées du *terrain*.

C_ENTIER(\$2;\$3;\$4;\$5)

` Vers un BLOB contenant des informations sur les *petits rectangles* une fois positionnés dans le *terrain*.

C_POINTEUR(\$6)

` Vers un BLOB contenant des informations sur les *petits rectangles* qui n'ont pas pu être placés dans le *terrain*.

C_POINTEUR(\$7)

Les BLOBs d'informations sont structurés ainsi : pour chaque *petit rectangle*, on trouve

dans l'ordre suivant, le numéro du rectangle, puis ses coordonnées gauche/haut/droit/bas. Chaque information est mise dans le BLOB en utilisant la commande **ENTIER VERS BLOB** dans l'Ordre octets macintosh.

Certains contraintes étaient imposées : interdit de retailler les rectangles, de les faire se chevaucher, de les faire pivoter. Le but du challenge était de disposer le plus possible de rectangles et il était clairement annoncé que le nombre de rectangles placés était plus important que la vitesse de placement.

Tous les challengers n'ont, hélas, pas expliqué leur algorithme et j'en profite pour remercier ceux qui l'ont fait. Cependant, les algorithmes partent du même principe : 1/ commencer par classer les petits rectangles afin d'éliminer ceux qui ne peuvent à l'évidence pas tenir, puis 2/ placer dans le terrain ceux qui ont survécu à ce classement. Le tri se faisant toujours par surface, mais pour certains, en tenant compte également de la hauteur et de la largeur des petits rectangles. En effet, un petit rectangle peut très bien avoir une surface minimale, mais être plus large que le terrain (exemple : un terrain de 100x100 pixels et un petit rectangle de 200x1 pixel).

Comme prévu dans l'énoncé, deux tests ont été réalisés. Pour coller un peu à une des idées de départ (organisation d'un calendrier d'occupation d'une salle de réunion), le second consistait à placer des petits rectangles de largeur identique dans un terrain plus haut que large (représentant par exemple une journée). Ce test était trop simple : tous les challengers ont placé le même nombre de rectangles ! Notez que les tests ont eu lieu avec 4D 673, structure compilée, système Windows NT, PC Athlon 700 MHz.

Voici les résultats. Les temps sont donnés en millisecondes, la base qui a servi à faire les tests et qui contient les codes de tous les challengers est sur le site ftp de *Planète 4D*. J'y ai ajouté une interface clignotante en cas de chevauchement afin que les erreurs éventuelles sautent aux yeux.

Test 1 (45 rectangles)

	Erreur	Rect.	Temps	Points
Hugues Gombert	Chevauchement	29	9,2	0
Jacques Fadeuilhe	-	32	11,4	32
Lackder Si-Youcef	-	30	11980	30
Silvain Traynard	-	29	980,4	29

Test 2 (35 rectangles)

	Erreur	Rect.	Temps	Points
Hugues Gombert	-	17	3,9	20
Jacques Fadeuilhe	-	17	5	19
Lackder Si-Youcef	-	17	44,6	18
Silvain Traynard	-	17	120,8	17

Totaux

1° Jacques Fadeuilhe	51 rectangles
2 Lackder Si-Youcef	48
3 Silvain Traynard	46
4 Hugues Gombert	20

Bravo à Jacques, qui remporte ce premier challenge de la nouvelle saison ! Le code de Lackder est très lent dans le premier test, mais c'est sans doute le temps nécessaire à la réflexion pour placer plus de rectangles que Silvain !

Au terme de ce premier challenge de la saison, Jacques possède 25 points, Lackder 15, Silvain 10 et Hugues 6. À noter que si un 5^{ème} challenger avait participé en laissant simplement son code vide, il serait arrivé 5^{ème} et aurait ainsi eu 3 points !

Voici le code qui a permis à Jacques de gagner.

```

` OrganizeRectangles - P4D Challenge #1
` Challenger : Jacques FADEUILHE
C_POINTEUR($1)
C_ENTIER($2;$3;$4;$5)
C_POINTEUR($6)
C_POINTEUR($7)

C_ENTIER($i;$j;$k;$iMin;$PMin;$NbRect;$T_Larg)
C_ENTIER('$T_Haut;$Num;$PX;$Aeration)
C_ENTIER LONG($Offset;$T_S;$Tot_S;$Reste_S;$NbMilliStart)
C_BOOLEEN($Terminé;$FirstLine;$Debug)

` Permet de tracer l'execution...
$Debug:=Macintosh control enfoncee | Windows Ctrl enfoncee

Si ($Debug)
  $NbMilliStart:=Nombre de millisecondes
  Creer fenetre(100;100;300;500;2)
  MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ " Init des
    variables."+Caractere(13))
Fin de si

` Init des variables de travail
$T_Larg:=($4-$2)
$T_Haut:=($5-$3)
$T_S:=$T_Larg*$T_Haut

$Aeration:=1
` Pour maîtriser l'occupation de l'espaaaaaaaace
TABLEAU ENTIER($tFrange;$T_Larg)

```

```

Si ($Debug)
  MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ " Récup
    Rect."+Caractere(13))
Fin de si

  `*** Récupérer les rectangles
  $NbRect:=Taille BLOB($1->)/10
  TABLEAU ENTIER($tRec_Num;$NbRect)
  TABLEAU ENTIER($tRec_G;$NbRect)
  TABLEAU ENTIER($tRec_H;$NbRect)
  TABLEAU ENTIER($tRec_D;$NbRect)
  TABLEAU ENTIER($tRec_B;$NbRect)
  TABLEAU ENTIER LONG($tRec_S;$NbRect)
  TABLEAU ENTIER($tRec_Larg;$NbRect)
  TABLEAU ENTIER($tRec_Haut;$NbRect)
  TABLEAU ENTIER($tRec_X;$NbRect)
  TABLEAU ENTIER($tRec_Y;$NbRect)
  $Offset:=0
  $Tot_S:=0
  $i:=1
  Tant que ($i<=$NbRect)
    $tRec_Num{$i}:=BLOB vers entier($1->;Ordre octets Macintosh;$Offset)
    $tRec_G{$i}:=BLOB vers entier($1->;Ordre octets Macintosh;$Offset)
    $tRec_H{$i}:=BLOB vers entier($1->;Ordre octets Macintosh;$Offset)
    $tRec_D{$i}:=BLOB vers entier($1->;Ordre octets Macintosh;$Offset)
    $tRec_B{$i}:=BLOB vers entier($1->;Ordre octets Macintosh;$Offset)
    $tRec_Larg{$i}:=$tRec_D{$i}-$tRec_G{$i}
    $tRec_Haut{$i}:=$tRec_B{$i}-$tRec_H{$i}
    $tRec_S{$i}:=$tRec_Larg{$i}*$tRec_Haut{$i}
    $Tot_S:=$Tot_S+$tRec_S{$i}
    `À Placer
    $tRec_X{$i}:=-1
    $tRec_Y{$i}:0

    `*** Eliminer tout de suite ce qui doit l'être !
    `Ah non, trop grand !
    Si (($tRec_Larg{$i}>$T_Larg) | ($tRec_Haut{$i}>$T_Haut))
      Si ($Debug)
        MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ "
          "+Chaine($tRec_Num{$i})+" Tient pas !" +Caractere(13))
      Fin de si
      `Mettre dans le Blob de sortie
      ENTIER VERS BLOB($tRec_Num{$i};$7->;Ordre octets Macintosh;)
      ENTIER VERS BLOB(0;$7->;Ordre octets Macintosh;)
      ENTIER VERS BLOB($tRec_H{$i};$7->;Ordre octets Macintosh;)
      ENTIER VERS BLOB($tRec_Larg{$i};$7->;Ordre octets Macintosh;)
      ENTIER VERS BLOB($tRec_B{$i};$7->;Ordre octets Macintosh;)

```

```

`Dégager les tableaux !
SUPPRIMER LIGNES($tRec_Num;$i;1)
SUPPRIMER LIGNES($tRec_G;$i;1)
SUPPRIMER LIGNES($tRec_H;$i;1)
SUPPRIMER LIGNES($tRec_D;$i;1)
SUPPRIMER LIGNES($tRec_B;$i;1)
SUPPRIMER LIGNES($tRec_S;$i;1)
SUPPRIMER LIGNES($tRec_Larg;$i;1)
SUPPRIMER LIGNES($tRec_Haut;$i;1)
SUPPRIMER LIGNES($tRec_X;$i;1)
SUPPRIMER LIGNES($tRec_Y;$i;1)
$NbRect:=$NbRect-1
Sinon
  $i:=$i+1
Fin de si

Fin tant que

`*** Les trier du plus petit au plus grand, pour en mettre le plus possible !
TRIER
  TABLEAU($tRec_S;$tRec_Num;$tRec_G;$tRec_H;$tRec_D;$tRec_B;$tRec_Larg;$tRec
    _Haut;$tRec_X;$tRec_Y;>)
  `Il y en a trop, c'est certain !
Si ($Tot_S>$T_S)
Si ($Debug)
  MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ " marquer ceux en
    trop."+Caractere(13))
Fin de si
$Tot_S:=0
$j:=$NbRect
Boucle ($i;1;$NbRect)
  $Tot_S:=$Tot_S+$tRec_S{$i}
Si ($Tot_S>$T_S)
  $j:=$i `
  $i:=$NbRect+1
Fin de si
Fin de boucle
  ` Ignorer ceux en trop !
Boucle ($i;$j;$NbRect)
  $tRec_X{$i}:=-2
Fin de boucle
Fin de si

`*** DEBUT TRAVAIL !

Si ($Debug)
MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ " Début

```

travail."+Caractere(13))

Fin de si

TRIER

TABLEAU(\$tRec_Haut;\$tRec_Larg;\$tRec_Num;\$tRec_S;\$tRec_G;\$tRec_H;\$tRec_D;\$tRec_B;\$tRec_X;\$tRec_Y;<)

\$Terminé:=**Faux**

\$FirstLine:=**Vrai**

\$PX:=0

\$Reste_S:=\$T_S

Repete

 `Prochain rectangle à placer

\$Num:=**Chercher dans tableau**(\$tRec_X;-1;1)

Au cas ou

: (\$Num<1) `Y'en à plus ! hip hip hip !

\$Terminé:=**Vrai**

 `Pas de place pour lui ! Éliminé ! zut !

: (\$tRec_S{\$Num}>\$Reste_S)

Si (\$Debug)

MESSAGE(Chaine(Nombre de millisecondes-\$NbMilliStart;"000 000")+
 "+Chaine(\$tRec_Num{\$Num})+" trop gros !"+Caractere(13))

Fin de si

\$tRec_X{\$Num}:=-3

: (\$FirstLine)

Si (\$Debug)

MESSAGE(Chaine(Nombre de millisecondes-\$NbMilliStart;"000 000")+
 "+Chaine(\$tRec_Num{\$Num})+" sur Lgn 1..." +Caractere(13))

Fin de si

Si ((\$PX+\$tRec_Larg{\$Num}-1)<\$T_Larg)

\$tRec_X{\$Num}:=\$PX

\$tRec_Y{\$Num}:=0

\$PX:=\$PX+\$tRec_Larg{\$Num}

\$Reste_S:=\$Reste_S-\$tRec_S{\$Num}

Boucle (\$i;\$tRec_X{\$Num};\$PX-1)

 \$tFrange{\$i}:=\$tRec_Haut{\$Num}+\$Aeration

Fin de boucle

\$PX:=\$PX+\$Aeration

Sinon

 `Fin de la 1ère ligne, travaillons maintenant par largeur décroissante...

\$FirstLine:=**Faux**

TRIER

TABLEAU(\$tRec_Larg;\$tRec_Haut;\$tRec_S;\$tRec_Num;\$tRec_G;\$tRec_H;\$tRec_D;\$tRec_B;\$tRec_X;\$tRec_Y;>)

Fin de si

`Bon, on le met ou ?...

Sinon

Si (\$Debug)

MESSAGE(Chaine(Nombre de millisecondes-\$NbMilliStart;"000 000")+
"+Chaine(\$tRec_Num{\$Num})+" a placer..." +Caractere(13))

Fin de si

`Se placer à gauche

\$PX:=0

`Le plus haut possible

\$PMin:=0

\$iMin:=0

Boucle (\$i;1;\$tRec_Larg{\$Num})

Si (\$tFrange{\$i}>\$tFrange{\$iMin})

\$iMin:=\$i

Fin de si

Fin de boucle

Si (\$tRec_Num{\$Num}=3)

Fin de si

`Déplacer le rectangle vers la droite pour qu'il soit le plus haut possible

\$i:=\$tRec_Larg{\$Num}

Repeter

\$PX:=\$PX+1

\$i:=\$i+1

Si ((\$tFrange{\$PX-\$Aeration}<(\$tFrange{\$iMin}-3)) & (\$tFrange{\$i}<(\$tFrange{\$iMin}-3)))

\$k:=\$PX

`trouver le point le plus haut

Boucle (\$j;\$k;\$i)

Si (\$tFrange{\$j}>\$tFrange{\$k})

\$k:=\$j

Fin de si

Fin de boucle

`Allez, on prend !

Si (\$tFrange{\$k}<(\$tFrange{\$iMin}-3))

\$iMin:=\$k

\$PMin:=\$PX

Fin de si

Fin de si

Jusque (\$i>=\$T_Larg)

`Ca tient ?

Si ((\$tFrange{\$iMin}+\$tRec_Haut{\$Num})<=\$T_Haut)

\$tRec_X{\$Num}:=\$PMin

Boucle (\$i;\$tRec_X{\$Num};\$tRec_X{\$Num}+\$tRec_Larg{\$Num})

Si (\$tFrange{\$i}>\$tFrange{\$iMin})

\$iMin:=\$i

Fin de si

Fin de boucle

\$tRec_Y{\$Num}:=\$tFrange{\$iMin}

\$Reste_S:=\$Reste_S-\$tRec_S{\$Num}

Boucle (\$i;\$tRec_X{\$Num};\$tRec_X{\$Num}+\$tRec_Larg{\$Num}-1)

\$tFrange{\$i}:=\$tRec_Y{\$Num}+\$tRec_Haut{\$Num}+\$Aeration

Fin de boucle

`Trop haut, éliminé !

Sinon

\$tRec_X{\$Num}:=-3

Fin de si

Fin de cas

Jusque (\$Terminé)

`*** FIN TRAVAIL

Si (\$Debug)

MESSAGE(Chaine(Nombre de millisecondes-\$NbMilliStart;"000 000")+"

Rangements"+Caractere(13))

Fin de si

`*** Ranger dans les Blobs de sortie

Boucle (\$i;1;\$NbRect)

Si (\$tRec_X{\$i}>=0) `Rectangle placé

ENTIER VERS BLOB(\$tRec_Num{\$i};\$6->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_X{\$i};\$6->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_Y{\$i};\$6->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_X{\$i}+\$tRec_Larg{\$i};\$6->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_Y{\$i}+\$tRec_Haut{\$i};\$6->;Ordre octets Macintosh;)*)

`Rectangle non placé => Le restituer intact :

Sinon

ENTIER VERS BLOB(\$tRec_Num{\$i};\$7->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_G{\$i};\$7->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_H{\$i};\$7->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_D{\$i};\$7->;Ordre octets Macintosh;)*)

ENTIER VERS BLOB(\$tRec_B{\$i};\$7->;Ordre octets Macintosh;)*)

Fin de si

Fin de boucle

`*** Effacer les Tableaux

TABLEAU ENTIER(\$tRec_Num;0)

TABLEAU ENTIER(\$tRec_G;0)

TABLEAU ENTIER(\$tRec_H;0)

TABLEAU ENTIER(\$tRec_D;0)

```

TABLEAU ENTIER($tRec_B;0)
TABLEAU ENTIER LONG($tRec_S;0)
TABLEAU ENTIER($tRec_Larg;0)
TABLEAU ENTIER($tRec_Haut;0)
TABLEAU ENTIER($tRec_X;0)
TABLEAU ENTIER($tRec_Y;0)

Si ($Debug)
MESSAGE(Chaine(Nombre de millisecondes-$NbMilliStart;"000 000")+ " Terminé
    !" + Caractere(13))
MESSAGE(Caractere(13)+ "Cliquez pour fermer !")
Repeter
POSITION SOURIS($i;$j;$k)
ENDORMIR PROCESS(Numero du process courant;10)
Jusque ($k>0)
FERMER FENETRE
Fin de si

`*** MàJ de OK en sortant...
OK:=Num(Taille BLOB($7->)=0)

```

<Encadré 1>

Petit règlement

- 1/ Il n'y a rien à gagner, si ce n'est être déclaré le *meilleur* à la fin de la saison.
- 2/ En participant, le challenger accepte que son code soit publié par les *Éditions des Sources* et mis en libre téléchargement sur le site web de *Planète 4D*. Tout cela gratuitement, bien entendu.
- 3/ Pour gagner un challenge, rien de plus simple : il faut non seulement réussir l'épreuve, mais la réussir plus vite que les autres. Le challenge Planète 4D est le seul endroit, avec les triggers, où l'on se doit (sauf énoncé contraire) de traquer les millisecondes. Les seuls plug-ins utilisables (si le challenger en éprouve le besoin) sont ceux fournis en standard par 4D dans son édition de base : 4D Chart, 4D Pack et InternetCommands. Aucun autre plug-in n'est autorisé, sauf si l'énoncé le précise clairement. Enfin, un challenge est toujours testé compilé, avec une version récente de 4D (le prochain challenge sera corrigé avec 4D 6.8.x). Si le code envoyé ne compile pas, le fichier d'erreur est renvoyé au challenger pour qu'il fasse la correction nécessaire (sauf si celle-ci prend très peu de temps).
- 4/ Un time out est imposé. Si un code met plus de 5 minutes à s'exécuter, il sera considéré comme étant entré dans une boucle sans fin. Sauf si l'énoncé autorise explicitement une durée différente.

</Encadré 1>

<Encadré 2>

Gagner des points

1/ Résoudre un challenge :

25 points pour le premier

15 pour le second

10 pour le troisième

6 pour le quatrième

3 pour le cinquième

2/ Proposer un challenge retenu (2 points). Nous sommes preneurs d'idées de challenge. Les contraintes sont les suivantes : il ne faut pas que l'on passe huit jours à les corriger tellement ils sont compliqués, et il ne faut pas laisser de place à la subjectivité (en se basant sur la beauté de telle ou telle interface des formulaires par exemple).

- Pour gagner un challenge, rien de plus simple : il faut non seulement réussir l'épreuve, mais la réussir plus vite que les autres. Le challenge Planète 4D est le seul endroit, avec les triggers, où l'on se doit de traquer les millisecondes ! Les seuls plug-ins utilisables (si le challenger en éprouve le besoin) sont ceux fournis en standard par 4D dans son édition de base : 4D Chart, 4D Pack et InternetCommands. Aucun autre plug-in n'est autorisé, sauf si l'énoncé le précise clairement. Enfin, un challenge est toujours testé compilé, avec la dernière version de 4D (en l'occurrence une version 6.7.x à ce jour). Si le code envoyé ne compile pas, le fichier d'erreur est renvoyé au challenger pour qu'il fasse la correction nécessaire (sauf ci celle-ci prend très peu de temps).

</Encadré 2>

<Encadré 3>

Mailing list pour le challenge

Une « mini-mailing list » a été mise en place pour cette saison de challenge. Seuls les abonnés de cette liste recevront les informations sur le challenge en cours : précisions sur l'énoncé, bugs de la base de test, etc. L'énoncé du challenge sera envoyé à tous les abonnés de la liste en même temps, permettant de niveller les problèmes de distribution du courrier.

Un mail posté sur la liste n'est pas redistribué aux autres abonnés de la liste.

Abonnez-vous en envoyant un mail à challenge@planete4d.com, en mettant « suscribe » comme sujet. Désabonnez vous en envoyant un mail dont le sujet est « unsubscribe ».

</encadré 3>

