`edt` : emploi du temps – version 0.6

David Cobac
(`dcobac@free.fr`)

Janvier 2007

Résumé

`edt` est un fichier de macros `metapost` permettant d'élaborer une image d'un emploi du temps. Il permet de réaliser facilement un tableau type, et est pourvu de nombreuses personnalisations possibles.

Table des matières

1 Installation et utilisation

1.1 Installation

`edt` n'est qu'un unique script `metapost`. Voici trois méthodes possibles pour l'installer (par ordre de préférence) :

- si vous avez un *texmf* local, copier le script seul `edt.mp` dans le répertoire `~/texmf/metapost/misc`. Ensuite il suffit de rafraîchir la base avec *texhash* par exemple ;
- vous pouvez préférer un emplacement qui vous est propre, il faudra alors ajouter le chemin où vous avez mis le script *edt.mp* dans la variable d'environnement `MPINPUTS` ;

– vous pouvez ne vouloir qu'essayer `edt`, vous placez alors le script dans le répertoire où vous comptez faire votre fichier `metapost` d'emploi du temps.

1.2 Utilisation

Le script est à utiliser comme n'importe quel fichier de macros `metapost`, il suffit donc pour l'utiliser d'invoquer :

```
input edt;
```

Le script lui-même invoque les fichiers de macros `boxes` et `string` normalement livré dans toute distribution `metapost`.

2 Exemple minimal

2.1 Code d'un emploi du temps

Voici ci-dessous un exemple relativement simple d'un fichier d'utilisation de `edt`. On peut observer assez immédiatement que le script se décompose en deux parties :
– la déclaration de l'emploi du temps lui-même ;
– les commandes de tracés encadrés par les marqueurs `beginfig` et `endfig`.
La partie déclarative comporte quatre parties :
– spécificités éventuelles de l'établissement (durée de la pause de midi, travail le mercredi après-midi etc.) ;
– définition des jours de travail ;
– définition des différents cours d'une semaine ;
– définition des créneaux des différents cours.

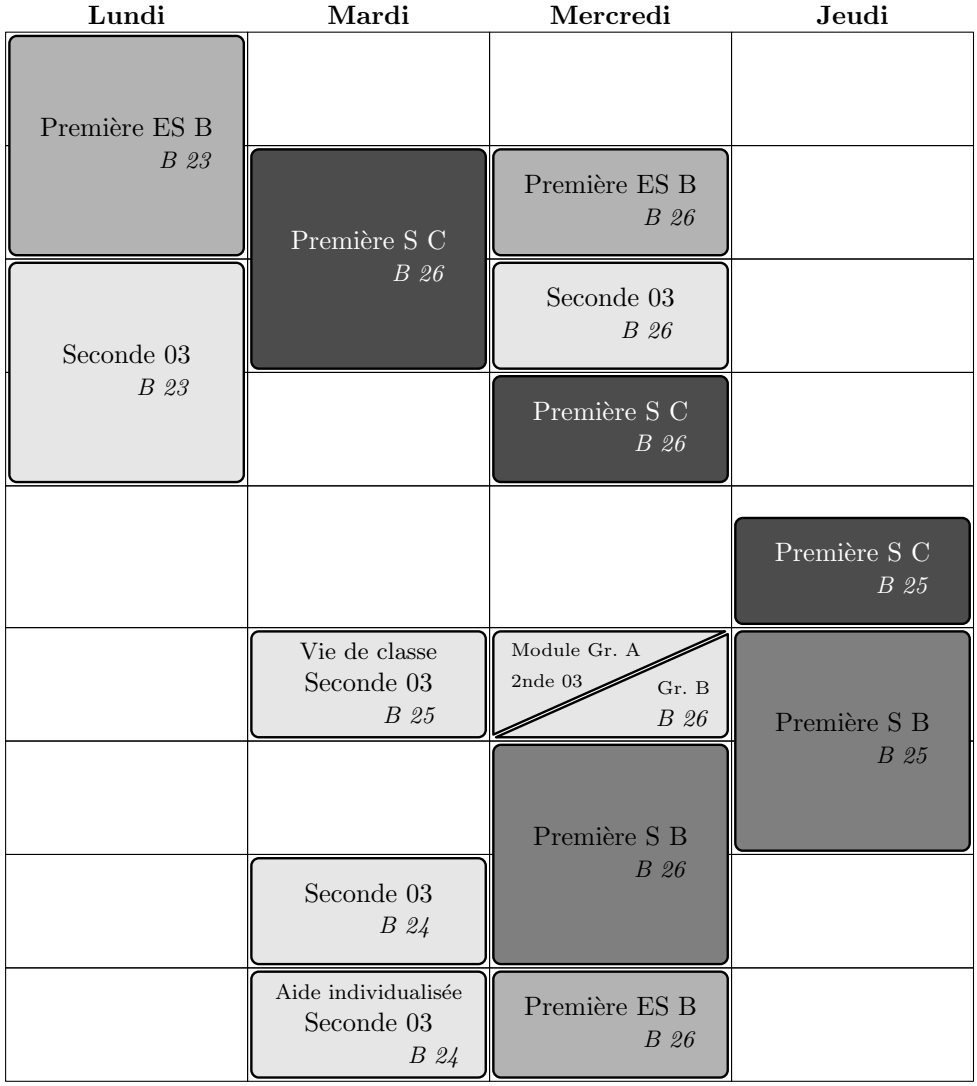
```
input edt;

%% établissement
dureeMidi:=1.25;
heuresPM:=4;
heuresPMMercredi:=4;
%% quadrillage de fond
jours(Lundi,Mardi,Mercredi,Jeudi);
%% identifiants des classes
classe(1)("Seconde 03")(noir,gris90,noir);
classe(2)("Première ES B")(noir,gris70,noir);
classe(3)("Première S B")(noir,gris50,noir);
classe(4)("Première S C")(blanc,gris30,noir);
classe(5)("\small Vie de classe\\ Seconde 03")(noir,gris90,noir);
classe(6)("\footnotesize Aide individualisée\\ Seconde 03")(noir,gris90,noir);
classe(7)("\scriptsize Module Gr. A\\\scriptsize 2nde 03")(noir,gris90,noir);
classe(8)("\scriptsize Gr. B")(noir,gris90,noir);
%% Lundi
cours(Lundi)(1,2)(2)("B 23");
cours(Lundi)(3,2)(1)("B 23");
%% Mardi
cours(Mardi)(2,2)(4)("B 26");
cours(Mardi)(6,1)(5)("B 25");
cours(Mardi)(8,1)(1)("B 24");
cours(Mardi)(9,1)(6)("B 24");
%% Mercredi
cours(Mercredi)(2,1)(2)("B 26");
cours(Mercredi)(3,1)(1)("B 26");
cours(Mercredi)(4,1)(4)("B 26");
coursQzH(Mercredi)(6,1)(7)("");
coursQzB(Mercredi)(6,1)(8)("B 26");
cours(Mercredi)(7,2)(3)("B 26");
cours(Mercredi)(9,1)(2)("B 26");
%% Jeudi
cours(Jeudi)(5.25,1)(4)("B 25");
cours(Jeudi)(6,2)(3)("B 25");

beginfig(0);
  traceFond;
  traceTout;
endfig;
end
```

2.2 Compilation

Voici le résultat de la compilation du fichier nommé `exemple1.mp` :

```
$ mpost exemple1.mp
This is MetaPost, Version 0.641 (Web2C 7.4.5)
(exemple1.mp (/home/david/texmf/metapost/misc/edt.mp
(/usr/share/texmf/metapost/base/boxes.mp)
(/usr/share/texmf/metapost/base/string.mp)) (mptextmp.mp) (mptextmp.mp)
(mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp)
(mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp)
(mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp) (mptextmp.mp)
(mptextmp.mp) (mptextmp.mp) [0] )
1 output file written: exemple1.0
Transcript written on exemple1.log.
```

2.3 Image finale



3 Dans l'ordre des choses

3.1 L'établissement

Les établissements ont des politiques très différentes sur les heures de travail, il faut donc éventuellement paramétrer la grille en fonction.

Par défaut, `edt` est paramétré pour prendre en compte 4 heures de travail le matin, 1 heure et demie de pause le midi et 3 heures de travail l'après-midi (mes besoins).

Pour modifier ces réglages, il suffit d'attribuer des valeurs différentes aux trois variables `heuresAM`, `dureeMidi` et `heuresPM`.

Par exemple :

```
dureeMidi=1.25;
heuresPM=4;
```

Par défaut, `edt` ne considère pas le mercredi après-midi, pour fixer des heures de travail durant cette période, par exemple 4 heures, il suffit de régler la variable `heuresPMMercredi` (de valeur nulle par défaut) :

```
heuresPMMercredi=4;
```

3.2 Définir les jours

Pour définir les jours de travail, il suffit d'invoquer la commande `jours` avec le nom des journées séparé par des virgules.

Cette commande est aussi celle qui trace la grille de fond, aussi faut-il préalablement (et éventuellement) inséré les commandes de personnalisations de la grille. La personnalisation la plus simple consiste à définir les dimensions des cellules de la grille ; par défaut une cellule a une longueur de 3 cm et une largeur de 1,5 cm.

```
largeurB=4cm;
hauteurB=2cm;
```

Les boîtes contenant le nom de la journée peuvent être personnalisées :

```
couleurBJ=vert;
couleurPJ=bleu;
encadreBJ=1;
```

signifie que la couleur de fond sera le vert, la couleur de police le noir et chaque jour sera encadré. Par défaut :

```
couleurBJ=blanc;
couleurPJ=noir;
encadreBJ=0;
```

La grille en elle-même peut être tracée avec toutes les options acceptées par `metapost` en utilisant la commande `optionsFond`, par exemple :

```
optionsFond:="withpen pencircle scaled .5mm dashed evenly withcolor red";
```

transmettra au traceur l'information de tracer la grille en pointillés rouge avec un crayon de pointe circulaire d'une épaisseur de 0,5 mm.

Finalement, on peut avoir quelquechose comme :

```
couleurBJ=vert;
couleurPJ=bleu;
encadreBJ=1;
optionsFond:="withpen pencircle scaled .5mm dashed evenly withcolor red";
jours(Lundi,Mardi,Mercredi,Jeudi);
```

3.3 Définir les classes

Définir une classe permet de lui associer :

- un identifiant numérique :1, 2 etc.
- un texte à afficher à chaque créneau de cette classe ;
- trois couleurs utilisés par les boîtes pour cette classe : une pour la police, une pour le fond et une pour le cadre.

On définira une classe à chaque fois qu'il faudra envisager un contenu texte différent, autrement dit une même classe (scolaire) peut faire définir plusieurs classes :

```
classe(1)("Seconde 03")(noir,gris90,noir);
...
classe(5)("\small Vie de classe\\ Seconde 03")(noir,gris90,noir);
classe(6)("\footnotesize Aide individualisée\\ Seconde 03")(noir,gris90,noir);
classe(7)("\scriptsize Module Gr. A\\\scriptsize 2nde 03")(noir,gris90,noir);
classe(8)("\scriptsize Gr. B")(noir,gris90,noir);
```

Bien sûr, on a choisi dans cet exemple, de conserver pour les mêmes couleurs pour pouvoir identifier visuellement qu'il s'agit bien de la même classe scolaire.

On insère du code LaTeX dans le texte à inscrire.

3.4 Définir les créneaux

Il ne reste plus qu'à entrer l'emploi du temps à proprement parler. `edt` fournit pour cela plusieurs commandes suivant la situation. Chacune d'elles produira un objet graphique aux couleurs et avec le texte de la classe choisie.

Ces objets sont des rectangles ou des triangles. On peut les choisir à coins arrondis avec la variable `arrondiB` (0 pour coins droits, 1 pour coins arrondis, par défaut), on peut aussi paramétrer la distance du bord avec le bord de la grille du fond avec la variable `offBord` valant 0,5 mm par défaut, et l'épaisseur du trait avec `epaisTrait` fixé par défaut à 1 bp. Par exemple :

```
arrondiB=0;
epaisTrait=.35mm;
offBord=0mm;
```

Une fois paramétré l'aspect des « boîtes », il suffit de définir les créneaux occupés. La grille de fond montre en fait tous les créneaux d'une journée, par exemple une journée de 4 heures le matin et 3 heures l'après-midi définit 9 créneaux, en effet, 1 créneau est ajouté pour la pause de midi. Ces créneaux sont numérotés de 1 à 9 dans l'ordre.

Par exemple, la définition de deux heures de cours le mardi à partie du 3^{e} créneau avec la classe numérotée 5 en salle « S 203 » sera :

```
cours(Mardi)(3,2)(5)("S 203");
```

Pour les cours de quinzaine, il est prévu une boîte triangulaire, elle peut-être haute ou basse, ainsi on pourra choisir pour une boîte haute :

```
coursQzH(Mardi)(3,2)(5)("S 203");
```

et pour une boîte basse :

```
coursQzB(Mardi)(3,2)(5)("S 203");
```

Le texte sera aligné à gauche pour les boîtes hautes, et à droite pour les boîtes basses.

4 Que va-ton afficher ?

Nous arrivons maintenant dans la partie traitant des commandes permettant de sortir une figure. Autrement dit, ce qui suit traite des commandes à placer entre les marqueurs `beginfig` et `endfig`.

4.1 Le fond : la grille

La grille est tracée avec la commande :

```
traceFond;
```

4.2 Tout

Pour tracer l'emploi du temps complet :

```
traceTout;
```

4.3 Une classe

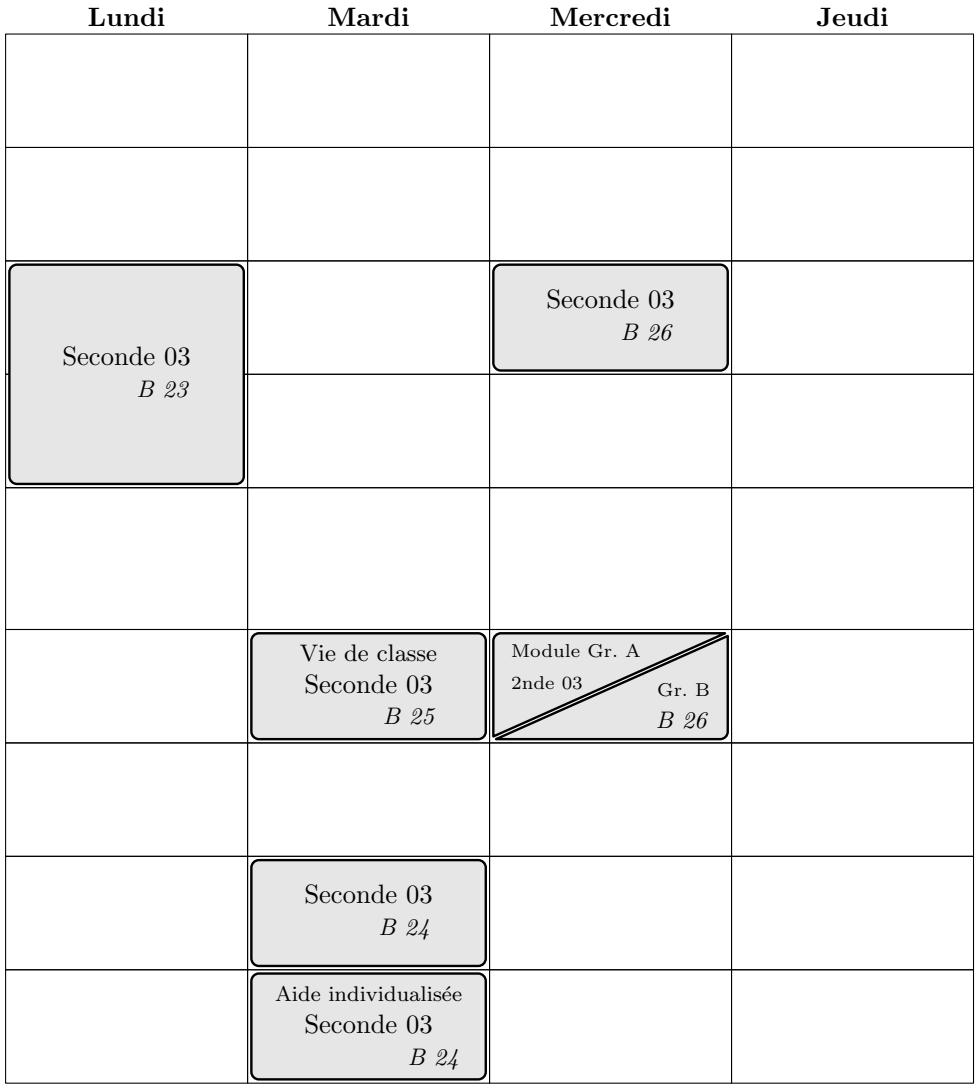
Pour tracer ce qui concerne la classe définie avec l'identifiant 1 :

```
traceClasse("1");
```

Dans notre exemple récurrent, la classe scolaire est définie par les classes 1, 5, 6, 7 et 8, on fera donc :

```
traceClasse("1 5 6 7 8");
```

qui nous donnera :



4.4 Une journée

Pour afficher le contenu d'une journée, par exemple le « Lundi », on invoque la commande :

```
traceJour(Lundi);
```

Pour réaliser 4 images correspondant aux jours de travail, il suffit de boucler sur le nom des journées et de réaliser les figures correspondantes :

```
j:=0;
forsuffixes $=Lundi,Mardi,Mercredi,Jeudi :
  beginfig(j);
    traceFond;
    traceJour($);
    j:=j+1;
  endfig;
endfor;
```

Voilà par exemple, l'emploi du temps de la journée de Mardi, obtenu avec :

```
traceJour(Mardi);
```

Lundi	Mardi	Mercredi	Jeudi
	Première S C *B 26*		
	Vie de classe Seconde 03 *B 25*		
	Seconde 03 *B 24*		
	Aide individualisée Seconde 03 *B 24*		

4.5 Une heure en évidence

Il peut être intéressant de mettre en évidence une heure de cours avec un autre style que celui défini. Pour cela, `edt` construit des doubles des images des heures de cours définis, ces doubles sont dessinés avec les variables de couleur `policeEvi`, `fondEvi` et `bordEvi`. Par défaut voici leurs valeurs :

```
policeEvi:=blanc;
fondEvi:=.5rouge;
bordEvi:=.2rouge;
```

On peut redéfinir les valeurs de ces variables dans la zone de déclaration **avant** de définir les cours.

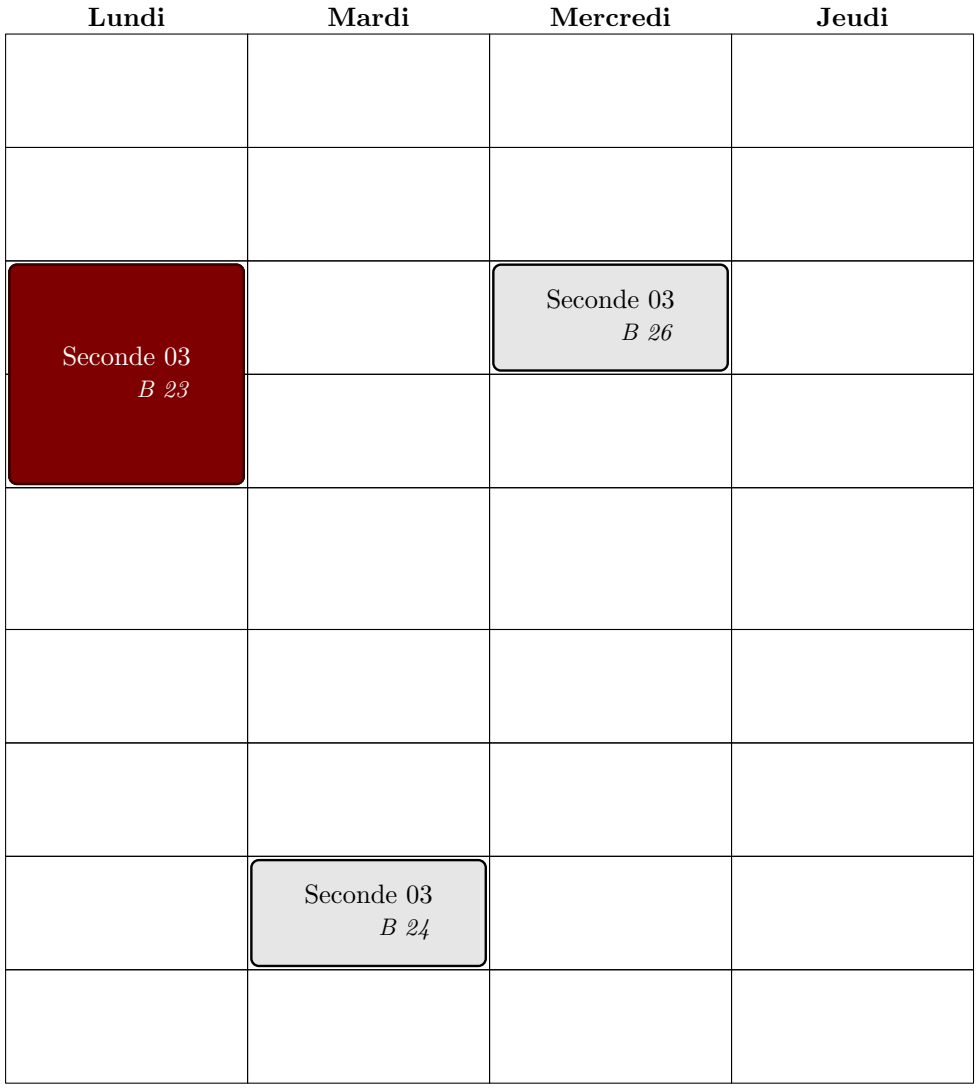
Pour tracer avec ces paramètres, il suffit d'invoquer la commande `traceHeureEvi` qui prend en argument le numéro d'ordre de l'heure de cours. Par exemple, si on veut mettre en évidence, dans notre exemple, l'heure de cours du lundi du troisième créneau horaire, il suffit de savoir que cette heure de cours a été défini en deuxième position, donc :

```
traceHeureEvi(2);
```

Si de plus on veut n'afficher que la classe 1 :

```
traceClasse("1");
traceHeureEvi(2);
```

On obtient :



4.6 Les horaires

On peut faire afficher les horaires d'entrée et de sortie de classe avec la commande `montreHoraires`. Son utilisation est plutôt fastidieuse mais elle a le mérite d'exister.

Elle prend comme argument TOUS les horaires et il faut les rentrer dans l'ordre ; chaque créneau nécessite deux horaires : la première heure est celle d'entrée en classe et la deuxième, celle de la sortie de classe.

Si vous avez défini 4 heures le matin, 3 heures l'après-midi, vous avez donc $4 + 1 + 3 = 8$ créneaux (1 pour l'heure du midi), vous allez avoir 16 arguments à la commande (en fait un seul argument, une chîne, contenant 16 horaires séparés par un espace).

Par exemple :

```
montreHoraires("8h15 9h10 9h12 10h07 10h22 11h17 11h20 12h15 pause rien 13h30 14h25 14h27 15h22 15h37 16h32 16h35 17h30");
```

On voit dans cet exemple que deux mots ont été inséré, le pause s'affichera effectivement, alors que « rien » est le mot-clé permettant de rien mettre à cet endroit, comme on peut le voir dans la section 5.

Les horaires s'afficheront à gauche de la colonne du lundi. En effet, pour l'instant il FAUT que Lundi (tel quel) soit une journée définie pour afficher les horaires.

On peut préciser la taille de la police avec l'option `taillePoliceHoraire` qui est fixée par défaut à `"footnotesize";`.

Par exemple :

```
taillePoliceHoraire:="\tiny";
```

On peut toujours positionner soi-même sans passer par la commande `montreHoraires`. Par exemple, pour indiquer « 10h07 » en bas à gauche du deuxième créneau du Lundi :

```
label.ulft(TEX("\footnotesize 10h07"),Lundi[2].sw);
```

4.7 Les couleurs

Pour simplifier la saisie, des couleurs sont prédéfinis :
- `noir`
- `blanc`
- 9 nuances de gris `gris10`, `gris20` etc.
- `rouge`
- `vert`
- `bleu`

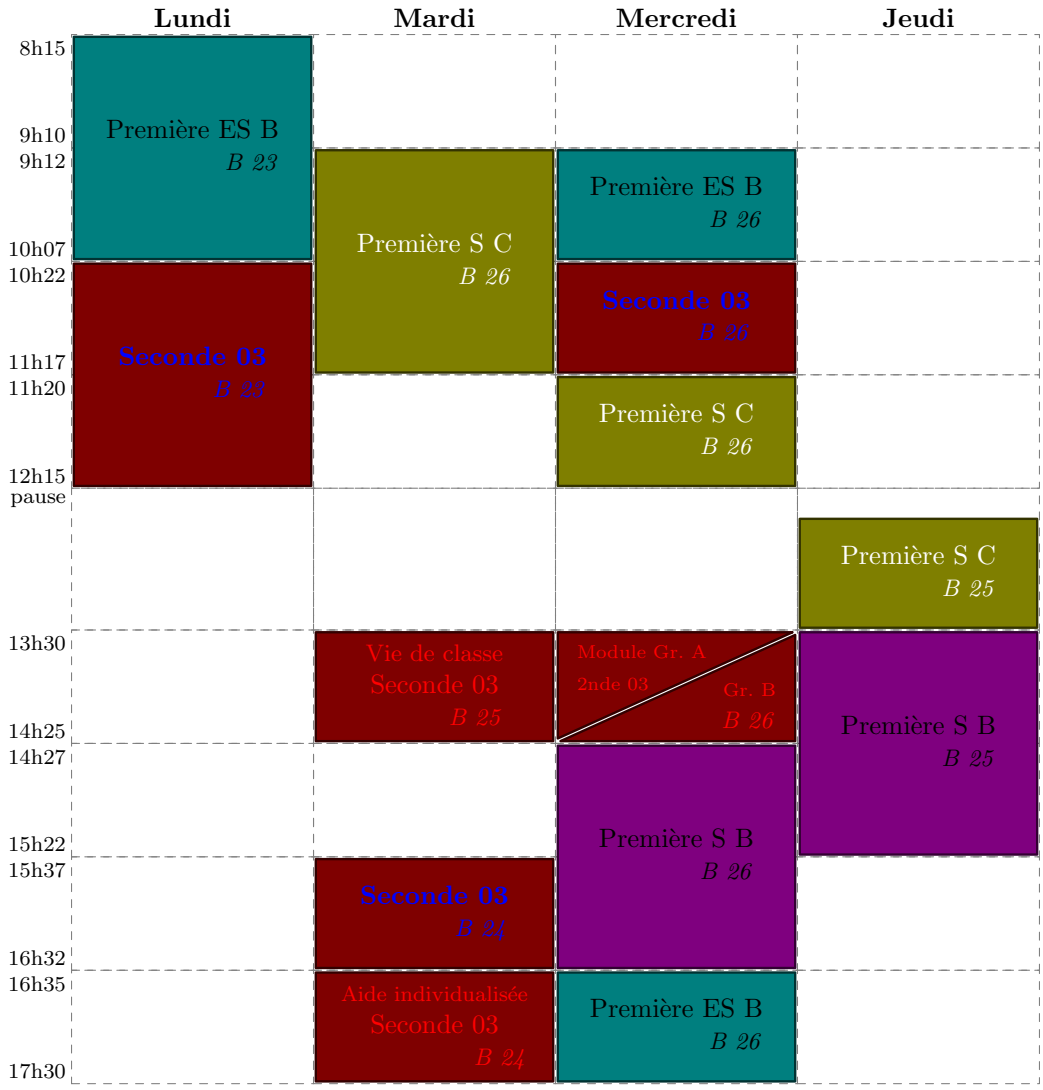
On peut évidemment les ajouter, soustraire etc. (c'est du `metapost`).

5 Le même exemple personnalisé

5.1 Code

```
input edt;

%% établissement
dureeMidi:=1.25;
heuresPM:=4;
heuresPMMercredi:=4;
%% quadrillage de fond
optionsFond:="withcolor gris50 dashed evenly";
jours(Lundi,Mardi,Mercredi,Jeudi);
montreHoraires("8h15 9h10 9h12 10h07 10h22 11h17 11h20 12h15 pause rien 13h30 14h25 14h27 15h22 15h37 16h32 16h35 17h30");
%% la personnalisation
offArrondi:=3mm;
offBord:=.3mm;
arrondiB:=0;
%% identifiants des classes
classe(1)("\bfseries Seconde 03")(bleu,.5rouge,.2rouge);
classe(2)("Première ES B")(noir,.5(bleu+vert),.2(bleu+vert));
classe(3)("Première S B")(noir,.5(bleu+rouge),.2(bleu+rouge));
classe(4)("Première S C")(blanc,.5(vert+rouge),.2(vert+rouge));
classe(5)("\small Vie de classe\\ Seconde 03")(rouge,.5rouge,.2rouge);
classe(6)("\footnotesize Aide individualisée\\ Seconde 03")(rouge,.5rouge,.2rouge);
classe(7)("\scriptsize Module Gr. A\\\scriptsize 2nde 03")(rouge,.5rouge,.2rouge);
classe(8)("\scriptsize Gr. B")(rouge,.5rouge,.2rouge);
%% Lundi
cours(Lundi)(1,2)(2)("B 23");
cours(Lundi)(3,2)(1)("B 23");
%% Mardi
cours(Mardi)(2,2)(4)("B 26");
cours(Mardi)(6,1)(5)("B 25");
cours(Mardi)(8,1)(1)("B 24");
cours(Mardi)(9,1)(6)("B 24");
%% Mercredi
cours(Mercredi)(2,1)(2)("B 26");
cours(Mercredi)(3,1)(1)("B 26");
cours(Mercredi)(4,1)(4)("B 26");
coursQzH(Mercredi)(6,1)(7)("");
coursQzB(Mercredi)(6,1)(8)("B 26");
cours(Mercredi)(7,2)(3)("B 26");
cours(Mercredi)(9,1)(2)("B 26");
%% Jeudi
cours(Jeudi)(5.25,1)(4)("B 25");
cours(Jeudi)(6,2)(3)("B 25");
beginfig(0);
  traceFond;
  traceTout;
endfig;
end
```

5.2 Image



6 Un PDF avec beamer

`beamer` permet d'inclure facilement des images issues de fichiers `metapost`. L'extension adéquate fournie avec `beamer` s'appelle `xmpmulti` et fournit la commande `\multiinput[<>]...+` Je laisse au lecteur le soin de compulser l'excellente et officielle documentation[1].

Néanmoins, cette solution n'a pas fonctionné chez moi, superposant sans remplacer les images. J'ai donc utiliser directement la commande `\includegraphics`

Voici le code produisant un PDF à partir des fichiers `exemple5.1` etc. :

```
\documentclass{beamer}
%% pour les images
\DeclareGraphicsRule{*}{mps}{*}{}
%%
\newcommand{\inclurefigure}[1]{%%
      \only<+>{%%
      \begin{center}%%
        \includegraphics[scale=.5]{#1}%%
      \end{center}%%
    }%%
}
%% début du document
\begin{document}
%
\begin{frame}
  \inclurefigure{exemple5.1}
  \inclurefigure{exemple5.2}
```

[1] La documentation de `beamer` ne dit pas à ce passage là que l'installation de `ConTeXt` est indispensable.

```
  \inclurefigure{exemple5.3}
  \inclurefigure{exemple5.4}
  \inclurefigure{exemple5.5}
  \inclurefigure{exemple5.6}
  \inclurefigure{exemple5.7}
  \inclurefigure{exemple5.8}
  \inclurefigure{exemple5.9}
  \inclurefigure{exemple5.10}
  \inclurefigure{exemple5.11}
  \inclurefigure{exemple5.12}
  \inclurefigure{exemple5.13}
  \inclurefigure{exemple5.14}
  \inclurefigure{exemple5.15}
\end{frame}
%
\end{document}
```

7 Image animée

On peut aisément réaliser des images animées. Par exemple, affichons successivement les heures de cours de la première à la dernière. Nous allons utiliser la commande `traceHeure` qui trace l'heure à la manière de `traceHeureEvi` mais avec le style originel. Nous allons donc à chaque figure, récupérer la figure précédente – sauf à la première figure qui va par contre dessiner le fond – et ajouter une heure. Le nombre d'heures qui a été défini par l'utilisateur est stockée dans la variable `numeroImage`.

Pour pouvoir utiliser nos polices dans le cadre d'une image, il est impératif d'attribuer une valeur différente de 0 à la variable `prologues`, on retrouvera donc une ligne `prologues:=1;` au début du script.

Voici donc le code obtenu :

```
input edt;

prologues:=1;

%% établissement
dureeMidi:=1.25;
heuresPM:=4;
heuresPMMercredi:=4;
%% quadrillage de fond
jours(Lundi,Mardi,Mercredi,Jeudi);
%% identifiants des classes
classe(1)("Seconde 03")(noir,gris90,noir);
classe(2)("Première ES B")(noir,gris70,noir);
classe(3)("Première S B")(noir,gris50,noir);
classe(4)("Première S C")(blanc,gris30,noir);
classe(5)("\small Vie de classe\\ Seconde 03")(noir,gris90,noir);
classe(6)("\footnotesize Aide individualisée\\ Seconde 03")(noir,gris90,noir);
classe(7)("\scriptsize Module Gr. A\\\scriptsize 2nde 03")(noir,gris90,noir);
classe(8)("\scriptsize Gr. B")(noir,gris90,noir);
%% Lundi
cours(Lundi)(1,2)(2)("B 23");
cours(Lundi)(3,2)(1)("B 23");
%% Mardi
cours(Mardi)(2,2)(4)("B 26");
cours(Mardi)(6,1)(5)("B 25");
cours(Mardi)(8,1)(1)("B 24");
cours(Mardi)(9,1)(6)("B 24");
%% Mercredi
cours(Mercredi)(2,1)(2)("B 26");
cours(Mercredi)(3,1)(1)("B 26");
cours(Mercredi)(4,1)(4)("B 26");
coursQzH(Mercredi)(6,1)(7)("");
coursQzB(Mercredi)(6,1)(8)("B 26");
cours(Mercredi)(7,2)(3)("B 26");
cours(Mercredi)(9,1)(2)("B 26");
%% Jeudi
cours(Jeudi)(5.25,1)(4)("B 25");
cours(Jeudi)(6,2)(3)("B 25");

%%
picture pre;
for j:=1 upto numeroImage :
  beginfig(j);
        if (j=1) : traceFond fi;
        if (j <> 1) : draw pre fi;
```

```
        traceHeure(j);
        pre:=currentpicture;
  endfig;
endfor;
end
```

Une fois les 15 images `.1` etc. `.15` produites, on renomme les 9 premières pour qu'elles aient des numéros `.01` `.02` etc, et on les isole dans un répertoire à part.

On passe alors le tout à la commande `convert` du logiciel `ImageMagick` :

```
$ convert exemple5.* exemple5.gif
```

On obtient alors un gif animé que l'on peut optimiser avec le logiciel `The Gimp` dans le menu *animation*. L'image perdra alors **beaucoup** de poids, c'est vraiment une étape indispensable.