

PixelArtTikz [fr]

Des PixelArts, en TikZ,
avec solution et couleurs.

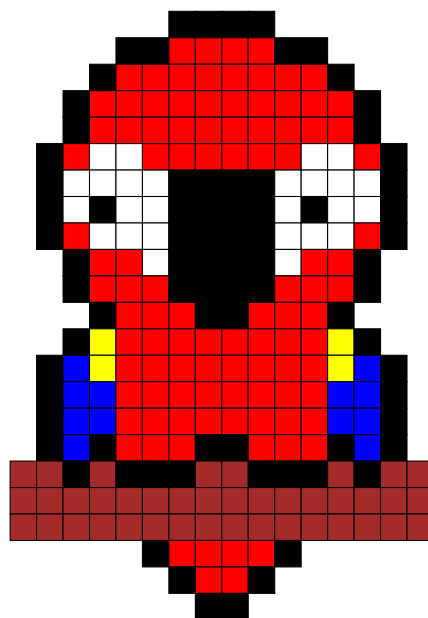
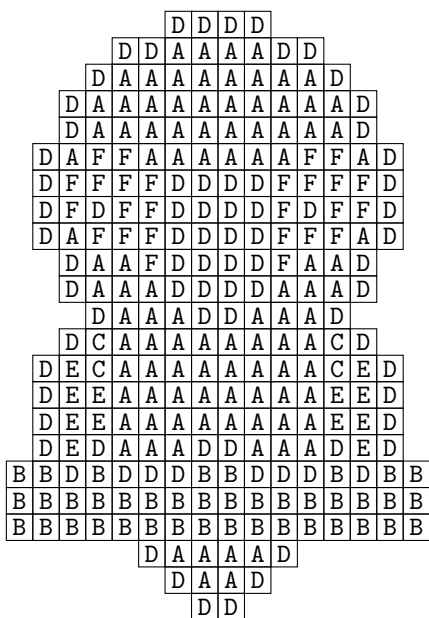
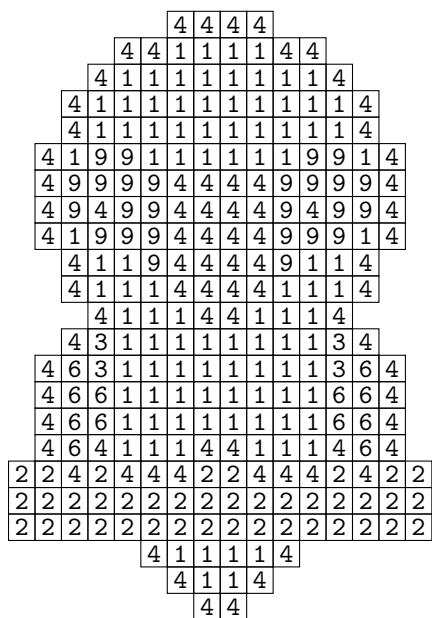
Version 0.1.0 - 23 Janvier 2023

Cédric Pierquet

c.pierquet - at - outlook . fr

<https://github.com/cpierquet/PixelArtTikz>

- Des commandes pour afficher des PixelArts.
- Environnement pour compléter éventuellement le PixelArt.



L^AT_EX

pdfL^AT_EX

LuaL^AT_EX

TikZ

T_EXLive

MiK_TE_X

Table des matières

I	Introduction	3
1	Le package PixelArtTikz	3
1.1	Introduction	3
1.2	Chargement du package, et option	3
1.3	Packages utilisés	3
1.4	Commandes et environnement	4
2	Petit aparté sur les fichiers csv	4
II	Commandes et environnement	5
3	La commande principale	5
3.1	Exemple introductif	5
3.2	Clés et options	6
3.3	Commande étoilée	9
4	Environnement PixelArt	10
4.1	Commande et options	10
4.2	Exemple	10
III	Historique	11

Première partie

Introduction

1 Le package PixelArtTikz

1.1 Introduction

L'idée est de *proposer*, dans un environnement TikZ, une commande permettant de générer des grilles PixelArt. Les données sont *lues* à partir d'un fichier csv, externe au fichier tex ou déclaré en interne grâce à l'environnement `filecontents`.

Avant toute chose, quelques petites infos sur les données au format csv, surtout dans l'optique de sa lecture et de son traitement par les commandes :

- le fichier de données csv doit être formaté avec le séparateur décimal « , » ;
- des cases vides seront codées par « - ».

Le fichier csv peut être déclaré directement dans le fichier tex, grâce à l'environnement `filecontents` (intégré en natif sur les dernières versions de L^AT_EX) :

```
\begin{filecontents*}{nomfichier.csv}
A,B,C,D
A,B,D,C
B,A,C,D
B,A,D,C
\end{filecontents*}
```

Code L^AT_EX

À la compilation, le fichier `nomfichier.csv` sera créé automatiquement, et l'option `{[overwrite]}` permet (logiquement) de propager les modifications au fichier csv.

1.2 Chargement du package, et option

Le package *central* est ici `csvsimple`, qui permet de lire et traiter le fichier csv.

Il est « disponible » en version L^AT_EX 2_ε ou en version L^AT_EX 3. Par défaut, PixelArtTikz le charge en version L^AT_EX 3, mais une option est disponible pour une *rétro-compatibilité* avec la version L^AT_EX 2_ε.

L'option `{[csvii]}` permet de passer l'appel au package en version L^AT_EX 2_ε.

```
\usepackage{PixelArtTikz}           %chargement du package version 3
%qui charge :
%\RequirePackage{expl3}
%\RequirePackage[13]{csvsimple}

\usepackage[csvii]{PixelArtTikz}    %chargement du package version 2
%qui charge :
%\RequirePackage[legacy]{csvsimple}
```

Code L^AT_EX

1.3 Packages utilisés

Le package est compatible avec les compilations usuelles en latex, pdf_latex, lua_latex ou xelatex.

Il charge les packages et bibliothèques suivantes :

- `tikz`, `xintexpr` et `xinttools` ;
- `xstring`, `xparse`, `simplekv` et `listofitems`.

1.4 Commandes et environnement

Il existe deux manières de représenter un PixelArt :

- soit par une commande autonome et indépendante ;
- soit par un environnement TikZ dans lequel du code pourra être *rajouté*.

Code \LaTeX

```
%Commande autonome
\PixelArtTikz[clés]<options tikz>{fichier.csv}

%Commande semi-autonome, à intégrer dans un environnement tikz
\PixelArtTikz*[clés]{fichier.csv}

%environnement
\begin{EnvPixelArtTikz}[clés]<options tikz>{fichier.csv}
  %code tikz
\end{EnvPixelArtTikz}
```

Concernant les couleurs, l'utilisateur utilisera celles disponibles avec les packages chargés.

La présente documentation a été compilée avec le package xcolor, chargé avec les options `{[table,svgnames]}`.

2 Petit aparté sur les fichiers csv

CSV désigne un format de fichiers dont le rôle est de présenter des données séparées par des virgules. Il s'agit d'une manière simplifiée d'afficher des données afin de les rendre transmissibles d'un programme à un autre.

Dans notre cas, le fichier csv contiendra les *codes* qui seront analysés un par un et ligne par ligne pour avoir le rendu par *code*, *symbole* ou *couleur*.

Il doit être préparé avec des caractères (codes) *simples* pour que le code de PixelArtTikz puisse fonctionner.

Deuxième partie

Commandes et environnement

3 La commande principale

3.1 Exemple introductif

La commande `\PixelArtTikz` nécessite de connaître :

- le fichier csv à traiter ;
- la liste (en fait sous forme de chaîne) des codes utilisés dans le fichier csv (comme 234679 ou ABCDJK...);
- la liste des symboles (éventuellement!) à afficher dans les cases s'il y a ambiguïté, comme 25,44,12 ou AA,AB,AC ;
- la liste des couleurs (si la correction est demandée), dans le même ordre que la liste des caractères.

On peut donc commencer par créer le fichier csv qui sera lu et interprété par les commandes du package. Le fichier peut-être créé directement dans la code du fichier tex.

Code \TeX

```
%déclaration du fichier csv
\begin{filecontents*}[overwrite]{base.csv}
  A,B,C,D
  A,B,D,C
  B,A,D,C
  C,A,B,D
\end{filecontents*}
```

Code \TeX

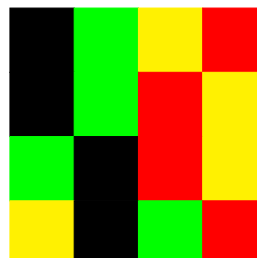
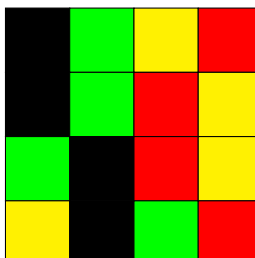
```
%notice et PixelArt
\begin{center}
  \begin{tblr}{colspec={*{4}{Q[1.25cm,c,m]}},hlines,vlines,rows={1.15em}}
    \SetCell[c=4]{c} Notice & & & \\
    A & B & C & D \\
    45 & 22 & 1 & 7 \\
    Noir & Vert & Jaune & Rouge \\
  \end{tblr}
\end{center}

\PixelArtTikz[Codes=ABCD,Style=\large\sffamily,Unite=0.85]{base.csv}
~~
\PixelArtTikz[Codes=ABCD,Symboles={45,22,1,7},Symb,Style=\large\sffamily,Unite=0.85]{base.csv}
~~
\PixelArtTikz[Codes=ABCD,Couleurs={black,green,yellow,red},Correction,Unite=0.85]{base.csv}
~~
\PixelArtTikz[Codes=ABCD,Couleurs={black,green,yellow,red},Correction,BordCases=false,Unite=0.85]{base.csv}
```

Notice			
A	B	C	D
45	22	1	7
Noir	Vert	Jaune	Rouge

A	B	C	D
A	B	D	C
B	A	D	C
C	A	B	D

45	22	1	7
45	22	7	1
22	45	7	1
1	45	22	7



3.2 Clés et options

```
\PixelArtTikz[clés]<options tikz>{fichier.csv}
```

Code \LaTeX

Le premier argument, *optionnel* et entre [...] propose des Clés nécessaires au bon fonctionnement de la commande :

- la clé **<Codes>** contient la *chaîne* des codes *simples* du fichier csv ;
- la clé **<Couleurs>** qui contient la *liste* des couleurs associées ;
- la clé **<Symboles>** qui contient la *liste éventuelles* des caractères alternatifs à afficher dans les cases ;
- la clé booléenne **<Correction>** qui permet de colorier le PixelArt ; défaut false
- la clé booléenne **<Symb>** qui permet d'afficher les caractères *alternatifs* ; défaut false
- la clé booléenne **<BordCases>** qui permet d'afficher les bords des cases de la correction ; défaut true
- la clé **<Style>** qui permet de spécifier le style des caractères. défaut scriptsize

Le second argument, *optionnel* et entre <...> sont des options – en langage TikZ – à passer à l'environnement qui sert de base au PixelArt.

Le troisième argument, *obligatoire*, est le nom du fichier csv à utiliser.

On rappelle que le fichier peut être créé au préalable, et placé dans le répertoire du fichier, ou bien il peut être créé *en direct*, à l'aide du package filecontents (chargé par \LaTeX).

```
%création du fichier csv
\begin{filecontents*}[overwrite]{test1.csv}
  --,--,-,--,-,4,4,4,4,-,--,-,--,-
  --,--,-,4,4,1,1,1,1,4,4,-,--,-
  --,--,-,4,1,1,1,1,1,1,1,4,-,--,-
  --,4,1,1,1,1,1,1,1,1,1,4,-,--,-
  --,4,1,1,1,1,1,1,1,1,1,1,4,-,--,-
  --,4,1,9,9,1,1,1,1,1,9,9,1,4,-
  --,4,9,9,9,9,4,4,4,4,9,9,9,4,-
  --,4,9,4,9,9,4,4,4,4,9,4,9,9,4,-
  --,4,1,9,9,9,4,4,4,4,9,9,9,1,4,-
  --,4,1,1,9,4,4,4,4,9,1,1,4,-,--
  --,4,1,1,1,4,4,4,4,1,1,1,4,-,--
  --,--,-,4,1,1,1,4,4,1,1,1,4,-,--,-
  --,--,-,4,3,1,1,1,1,1,1,1,1,3,4,-,--
  --,4,6,3,1,1,1,1,1,1,1,1,3,6,4,-
  --,4,6,6,1,1,1,1,1,1,1,1,6,6,4,-
  --,4,6,6,1,1,1,1,1,1,1,1,6,6,4,-
  --,4,6,4,1,1,1,4,4,1,1,1,4,6,4,-
  2,2,4,2,4,4,4,2,2,4,4,4,2,4,2,2
  2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2
  2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2
  --,--,-,--,-,4,1,1,1,1,4,-,--,-,--,-
  --,--,-,--,-,4,1,1,4,-,--,-,--,-,--
  --,--,-,--,-,4,4,-,--,-,--,-,--,-
\end{filecontents*}
```

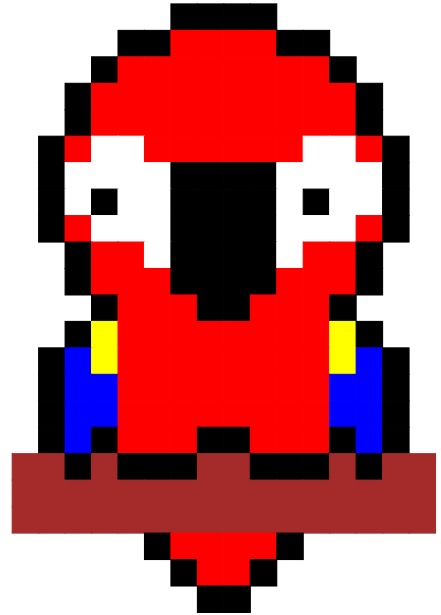
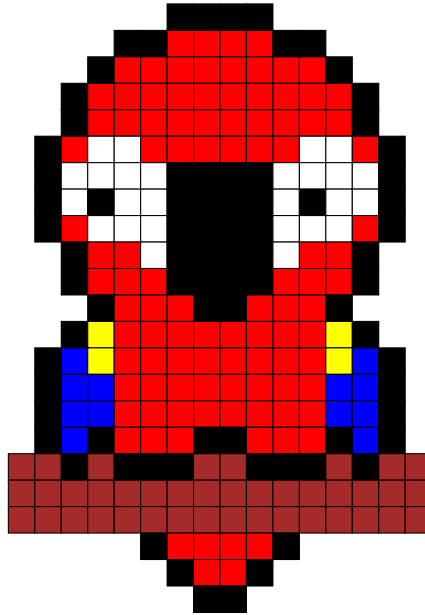
Code \LaTeX

```

%codes simples et sans ambiguïté
%une case vide sera codée par -
\PixelArtTikz[Codes=123469,Style=\ttfamily,Unite=0.35]{test1.csv}
~~
\PixelArtTikz[Codes=123469,Couleurs={Red,Brown,Yellow,Black,Blue,White},Correction,Unite=0.35]{test1.csv}
~~
\PixelArtTikz[Codes=123469,Couleurs={Red,Brown,Yellow,Black,Blue,White},Correction,Unite=0.35,BordCases=false]%
{test1.csv}

```

			4	4	4	4							
		4	4	1	1	1	1	4	4				
	4	1	1	1	1	1	1	1	1	4			
	4	1	1	1	1	1	1	1	1	1	4		
	4	1	1	1	1	1	1	1	1	1	4		
4	1	9	9	1	1	1	1	1	1	9	9	1	4
4	9	9	9	9	4	4	4	4	9	9	9	9	4
4	9	4	9	9	4	4	4	4	9	4	9	9	4
4	1	9	9	9	4	4	4	4	9	9	9	1	4
	4	1	1	9	4	4	4	4	9	1	1	4	
	4	1	1	1	4	4	4	1	1	1	4		
		4	1	1	1	4	4	1	1	1	4		
	4	3	1	1	1	1	1	1	1	3	4		
4	6	3	1	1	1	1	1	1	1	3	6	4	
4	6	6	1	1	1	1	1	1	1	6	6	4	
4	6	6	1	1	1	1	1	1	1	6	6	4	
4	6	4	1	1	1	4	4	1	1	4	6	4	
2	2	4	2	4	4	4	2	2	4	4	4	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2
			4	1	1	1	1	4					
			4	1	1	4							
			4	4									



3.3 Commande étoilée

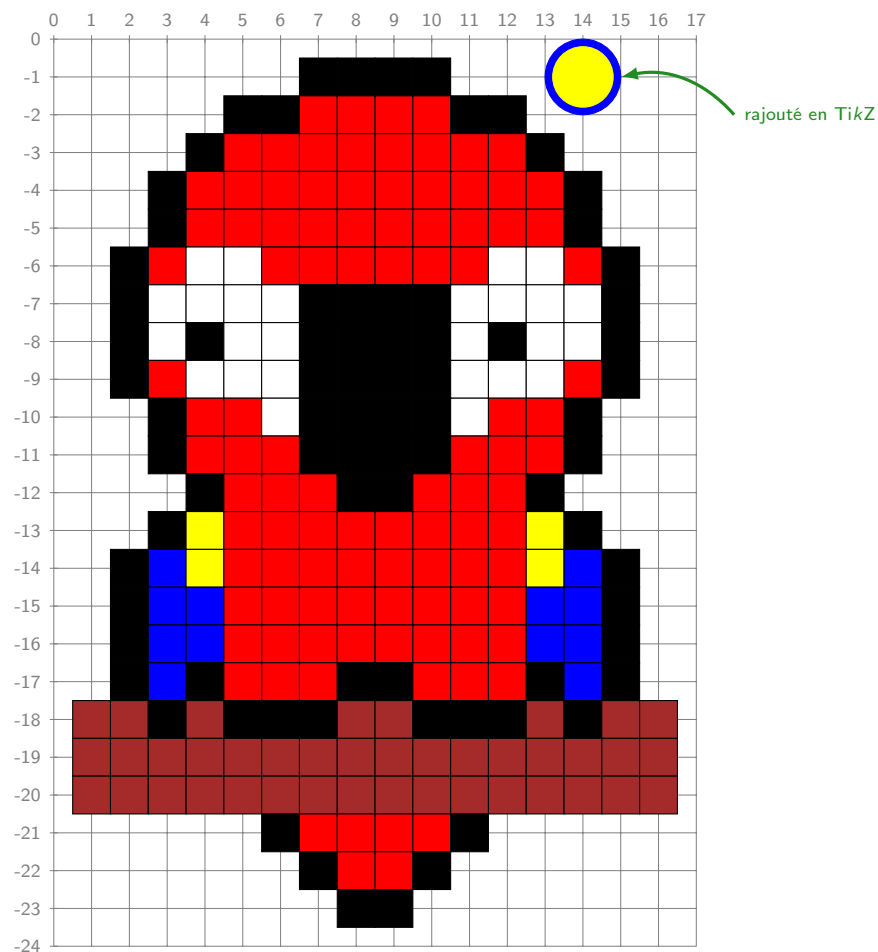
La commande étoilée `\PixelArtTikz*` permet d'intégrer le PixelArt dans un environnement créé par l'utilisateur. Cela permet par exemple de pouvoir rajouter du code en parallèle du PixelArt.

Il est à noter que, dans ce cas :

- l'argument *optionnel* entre `<...>` est inutile;
- la clé **(Unite)** n'intervient plus dans le tracé (elle peut être passée directement dans l'environnement !)

Code \LaTeX

```
\begin{center}
  \begin{tikzpicture}[scale=0.5]
    %grille pour visualiser le positionnement
    \draw[very thin,gray,xstep=1,ystep=1] (0,0) grid (17,-24) ;
    \foreach \x in {0,1,...,17} \draw[very thin,gray] (\x,-3pt)--(\x,3pt)%
    node[above,font=\scriptsize\sffamily] {\x} ;
    \foreach \y in {0,-1,...,-24} \draw[very thin,gray] (3pt,\y)--(-3pt,\y)%
    node[left,font=\scriptsize\sffamily] {\y} ;
    %le PixelArt
    \PixelArtTikz*[Codes=123469,Couleurs={Red,Brown,Yellow,Black,Blue,White},Correction]{test1.csv}
    %du code rajouté
    \filldraw[Blue] (14,-1) circle[radius=1] ;
    \filldraw[Yellow] (14,-1) circle[radius=0.8] ;
    \draw[ForestGreen,very thick,<-,>=latex] (15,-1) to[bend left=30] (18,-2)%
    node[right,font=\scriptsize\sffamily] {rajouté en Ti\textit{k}Z} ;
  \end{tikzpicture}
\end{center}
```



4 Environnement PixelArt

4.1 Commande et options

Le package `PixelArtTikz` propose également un environnement pour créer un `PixelArt`, et pouvoir rajouter des éléments en marge du `PixelArt`.

- L'environnement est créé autour de `TikZ` et le code rajouté le sera dans un langage `TikZ`!
- Le code rajouté le sera, dans ce cas, *au-dessus* du `PixelArt`!

Le fonctionnement global est le même que pour la commande autonome.

```
\begin{EnvPixelArtTikz}[clés]<options tikz>{fichier.csv}
  %code(s) tikz qui seront au-dessus du PixelArt
\end{EnvPixelArtTikz}
```

Code \LaTeX

Le premier argument, *optionnel* et entre `[...]` propose des Clés nécessaires au bon fonctionnement de la commande :

- la clé **<Codes>** contient la *chaîne* des codes *simples* du fichier `csv` ;
- la clé **<Couleurs>** qui contient la *liste* des couleurs associées ;
- la clé **<Symboles>** qui contient la *liste éventuelles* des caractères alternatifs à afficher dans les cases ;
- la clé booléenne **<Correction>** qui permet de colorier le `PixelArt` ; défaut false
- la clé booléenne **<Symb>** qui permet d'afficher les caractères *alternatifs* ; défaut false
- la clé booléenne **<BordCases>** qui permet d'afficher les bords des cases de la correction ; défaut true
- la clé **<Style>** qui permet de spécifier le style des caractères. défaut `scriptsize`

Le second argument, *optionnel* et entre `<...>` sont des options – en langage `TikZ` – à passer à l'environnement qui sert de base au `PixelArt`.

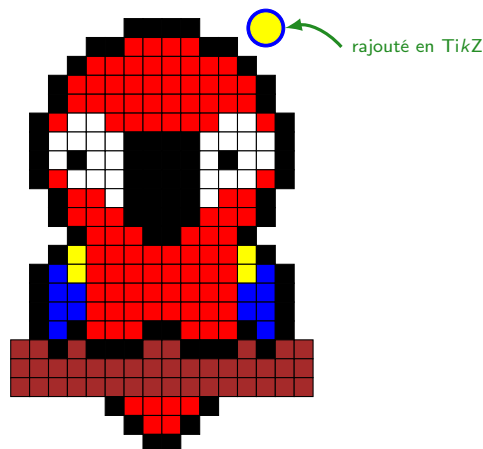
Le troisième argument, *obligatoire*, est le nom du fichier `csv` à utiliser.

4.2 Exemple

Les symboles affichés dans les cases sont situés aux nœuds de coordonnées $(c; -l)$ où l et c sont les numéros de ligne et de colonne correspondants à la position de la donnée dans le fichier `csv`.

```
\begin{center}
  \begin{EnvPixelArtTikz}%
    [Codes=123469,Couleurs={Red,Brown,Yellow,Black,Blue,White},Correction,Unite=0.25]
    {test1.csv}
    \filldraw[Blue] (14,-1) circle[radius=1] ;
    \filldraw[Yellow] (14,-1) circle[radius=0.8] ;
    \draw[ForestGreen,very thick,<-,>=latex] (15,-1) to[bend left=30] (18,-2)%
      node[right,font=\scriptsize\sffamily] {rajouté en Ti\textit{k}Z} ;
  \end{EnvPixelArtTikz}
\end{center}
```

Code \LaTeX



Troisième partie

Historique

v0.1.0 : Version initiale