PixelArtTikz [en]

PixelArts, with TikZ,
with solution and colors.

Version 0.1.2 - 12/10/2023

Cédric Pierquet
c pierquet – at – outlook . fr
https://github.com/cpierquet/PixelArtTikz

▶ Commands to display PixelArts.
▶ Environment to complete the PixelArt.
# Contents

## I Introduction  
1 The package PixelArtTikz
   1.1 Introduction ................................................................. 3  
   1.2 Loading of the package, and option .................................. 3  
   1.3 Used packages .............................................................. 3  
   1.4 Macros and environment ................................................... 4

## II Macros and environment  
3 Main macro
   3.1 Example ............................................................................ 5  
   3.2 Options and keys .............................................................. 6  
   3.3 Starred macro .................................................................. 9

## III History

4 PixelArt environment
   4.1 Usage ............................................................................... 10  
   4.2 Example ............................................................................ 10

5 Macro for mini-PixelArt
   5.1 Idea ............................................................................... 11  
   5.2 Examples .......................................................................... 11

III History  

12
Part I

Introduction

1 The package PixelArtTikz

1.1 Introduction

The idea is to propose, within a Ti\kZ environment, a macro to generate PixelArt. The data is read from a csv file, already existing in the folder of the tex file, or created on-the-fly by filecontents.

Some advices about the csv file :

- the csv file must use "," as separator;
- empty cells are coded by "-".

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

While compiling, the file filename.csv will be created, and the option (overwrite) will propagate the modifications!

1.2 Loading of the package, and option

The package csvsimple is necessary in order to read the csv file. The package is available in two versions, one written in \LaTeX\ and the other in \LaTeX3. By default, PixelArtTikz loads the \LaTeX3 version, but an option is available to work with the \LaTeX2\ε version.

The option ([csvii]) forces the usage of the \LaTeX2\ε version.

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

While compiling, the file filename.csv will be created, and the option (overwrite) will propagate the modifications!

1.3 Used packages

It’s fully compatible with usual \TeX engines, such as \LaTeX, pdflatex, lualatex or xelatex.

It loads the following packages and libraries:

- tikz, xintexpr et xinttools;
- xstring, xparse, simplekv and listofitems.
1.4 Macros and environment

There are two ways to create PixelArt:

- with an independent macro;
- with a TiKZ environment in order to add code afterwards.

\begin{EnvPixlArtTikz}
%tikz code
\end{EnvPixlArtTikz}

2 Colors

Concerning colors: the user can use all colors provided by loaded packages!
Without extra packages, the available colors are:

- magenta
- cyan
- blue
- green
- red
- darkgray
- olive
- lime
- brown
- lightgray
- white
- gray
- black
- yellow
- violet
- teal
- purple
- pink
- orange
Part II
Macros and environment

3 Main macro

3.1 Example

The macro \texttt{\texttt{PixlArtTikz}} needs:

- the file \texttt{csv};
- the list (by a string) of codes used in the file \texttt{csv} (e.g. 234679 or ABCDJK...);
- the list of symbols (if needed) to print in the cells, e.g. 25,44,12 or AA,AB,AC;
- the list of colors (for the correction), same order as the codes.

We can begin by creating the file \texttt{csv}, directly within the tex code, or with a external file.

\begin{verbatim}
%creation of the csv
\begin{filecontents*}[overwrite]{base.csv}
A,B,C,D
A,B,D,C
B,A,D,C
C,A,B,D
\end{filecontents*}
\end{verbatim}

\begin{verbatim}
%instructions and pixelarts
\begin{center}
\begin{tblr}{colspec={*{4}{Q[1.25cm,c,m]}},hlines,vlines,rows={1.15em}}
\SetCell[c=4]{c} Instructions & & & \\
A & B & C & D \\
45 & 22 & 1 & 7 \\
\end{tblr}
\end{center}
\PixlArtTikz[Codes=ABCD,Style=\large\sffamily,Unit=0.85]{base.csv}
\end{verbatim}

\begin{verbatim}
\begin{center}
\begin{tblr}{colspec={*{4}{Q[1.25cm,c,m]}},hlines,vlines,rows={1.15em}}
\SetCell[c=4]{c} Instructions & & & \\
A & B & C & D \\
45 & 22 & 1 & 7 \\
\end{tblr}
\end{center}
\PixlArtTikz[Codes=ABCD,Symbols={45,22,1,7},Symb,Style=\large\sffamily,Unit=0.85]{base.csv}
\end{verbatim}

\begin{verbatim}
\begin{center}
\begin{tblr}{colspec={*{4}{Q[1.25cm,c,m]}},hlines,vlines,rows={1.15em}}
\SetCell[c=4]{c} Instructions & & & \\
A & B & C & D \\
45 & 22 & 1 & 7 \\
\end{tblr}
\end{center}
\PixlArtTikz[Codes=ABCD,Colors={black,green,yellow,red},Correction,Unit=0.85]{base.csv}
\end{verbatim}

\begin{verbatim}
\begin{center}
\begin{tblr}{colspec={*{4}{Q[1.25cm,c,m]}},hlines,vlines,rows={1.15em}}
\SetCell[c=4]{c} Instructions & & & \\
A & B & C & D \\
45 & 22 & 1 & 7 \\
\end{tblr}
\end{center}
\PixlArtTikz[Codes=ABCD,Colors={black,green,yellow,red},Correction,Border=false,Unit=0.85]{base.csv}
\end{verbatim}
3.2 Options and keys

\PixlArtTikz\[keys\]<options tikz>{file.csv}

The first argument, *optional* and between \[\ldots\] proposes the keys:

- the key \textbf{(Codes)} with the \textit{string} of \textit{simple} codes of the \texttt{csv} file;
- the key \textbf{(Colors)} with the \textit{list} of colors;
- the key \textbf{(Symbols)} with the \textit{optional list} of alt. symbols for the cells;
- the boolean \textbf{(Correction)} to color the PixelArt; \texttt{default} \texttt{false}
- the boolean \textbf{(Symb)} to print the symbols; \texttt{default} \texttt{false}
- the boolean \textbf{(Border)} to print borders of the cells; \texttt{default} \texttt{true}
- the key \textbf{(Style)} to specify the style of the text. \texttt{default} \texttt{\scriptsize}

The second argument, *optional* and between <\ldots>, are \texttt{TikZ} options to pass on to the environment which creates the PixelArt.

The third argument, *mandatory*, is the filename of the \texttt{csv}.

\begin{verbatim}
%creation of the csv
\begin{filecontents*}[overwrite]{test1.csv}
-,-,-,-,-,-,4,4,4,4,-,-,-,-,-
-,-,-,4,1,1,1,1,4,4,-,-,-,-
-,-,-,4,1,1,1,1,1,1,4,-,-
-,-,4,1,1,1,1,1,1,1,1,4,-
-,-,4,1,1,1,1,1,1,1,1,4,-
-,-,4,1,1,1,1,1,1,1,1,4,-
-4,1,9,9,1,1,1,1,9,9,1,4,-
-4,9,9,9,9,4,4,4,9,9,9,4,-
-4,9,4,4,4,4,4,4,9,4,9,4,-
-4,1,9,9,9,4,4,4,9,9,9,4,-
-4,1,1,9,4,4,4,9,1,1,4,-
-4,1,1,4,4,4,4,1,1,1,4,-
-4,1,1,4,1,1,4,1,1,1,4,-
-4,1,1,1,1,1,1,1,1,3,4,-
-4,6,3,1,1,1,1,1,1,3,6,-
-4,6,6,1,1,1,1,1,1,6,6,-
-4,6,6,1,1,1,1,1,1,6,6,-
-4,6,4,1,1,1,4,1,1,1,6,6,-
2,2,4,2,4,4,4,2,4,4,4,2,2
2,2,2,2,2,2,2,2,2,2,2,2
2,2,2,2,2,2,2,2,2,2,2,2
-,-,-,-,-,-,4,1,1,4,4,-,-,-,-
-,-,-,-,-,-,4,1,1,4,4,-,-,-,-
-,-,-,-,-,-,4,1,1,4,4,-,-,-,-
\end{filecontents*}
\end{verbatim}
\texttt{\textbackslash PixlArtTikz[Codes=123469,Style=\texttt{ttfamily},Unit=0.35]{test1.csv}}

\texttt{\textbackslash PixlArtTikz[Codes=123469,Colors=\{red,brown,yellow,black,blue,white\},Correction,Unit=0.35]{test1.csv}}

\texttt{\textbackslash PixlArtTikz[Codes=123469,Colors=\{red,brown,yellow,black,blue,white\},Correction,Unit=0.35,Border=false]{test1.csv}}
In the following example, the *symbols* to print can’t be used for the *codes*, so we can use the keys *(Symbols)* and *(Symb)* to bypass this limitation.
3.3 Starred macro

The starred macro \PixlArtTikz* is to be used within an already created environment. It can be useful for adding code after the PixelArt.

In this case:

- the optional argument between <...> is discarded;
- the key \texttt{(Unit)} is discarded too (units can be configured in the environment!)

```latex
\begin{center}
\begin{tikzpicture}[scale=0.5]
%grid to show positioning
\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
\PixlArtTikz*[Codes=123469,Colors={red,brown,yellow,black,blue,white},Correction]{test1.csv} %added code
\filldraw[blue] (14,-1) circle[radius=1];
\filldraw[yellow] (14,-1) circle[radius=0.8);\draw[green,very thick,<-latex] (15,-1) to[bend left=30] (18,-2);\node[right,font=\scriptsize\sffamily] {Code Ti\textit{k}Z};
\end{tikzpicture}
\end{center}
```
4 PixelArt environment

4.1 Usage

The package PixelArtTikz provides an environment to create a PixelArt and add code afterwards.

- The environment is created within TikZ and additional code is passed on to the TikZ environment!
- The additional code will be printed on top of the PixelArt!

\begin{EnvPixlArtTikz}\{keys\}<options tikz>{filename.csv}
\%tikz code(s)
\end{EnvPixlArtTikz}

The first argument, *optional* and between [...], proposes the keys:

- the key (Codes) with the *string of simple codes* of the csv file;
- the key (Colors) with the *list of colors*;
- the key (Symbols) with the *optional list of alt. symbols* for the cells;
- the boolean (Correction) to color the PixelArt; default false
- the boolean (Symb) to print the symbols; default false
- the boolean (Border) to print borders of the cells; default true
- the key (Style) to specify the style of the text. default \scriptsize

The second argument, *optional* and between <...>, is for TikZ options to be passed on to the environment which creates the PixelArt.

The third argument, *mandatory*, is the filename of the csv.

4.2 Example

The symbols are at the nodes (c; −l) where l and c are the row and column of the data in the csv file.
5 Macro for *mini*-PixelArt

5.1 Idea

The idea is to propose a macro to insert, without csv file, a small PixelArt with small colors list.

\MiniPixlArt[{keys}][list of colors]

The first argument, *optional* and between [*...*] proposes the *keys*:

- the key *(Unit)* for dimension of the cells ; default 0.25em,
- the boolean *(Border)* to print a small border for the cells. default false

The second argument, *mandatory* and between {*...*}, is the colors of the cells :

- each color is *coded* by a letter :
  - R : red  
  - C : blue  
  - B : black  
  - . : white  
  - G : green  
  - Y : yellow  
  - L : gray  
  - M : maroon  
  - O : orange  
  - P : purple

- each linebreak is done by , ;
- the thickness of the borders are 10% of the unit.

The last argument, *optional* and between <...>, proposes options for the tikz environment.

5.2 Examples

\MiniPixlArt{{% ..RR...RR..  .RRRRRRRR..  RRRRRRRRRR  RRRRRRRRRRR  RRRRRRRRRRR  RRRRRRRRRRRR  .RRRRRRRRRR...  ...RRRRRRRRR...  ....RRRRRRRRRRR  ....RRRRRRRRRRRR  ....RRRRRRRRRRRRR  \}}

Inline, we can give \MiniPixlArt[Unit=5mm,Border]{BCGOYG,YLP.BR}<baseline=(current bounding box.center)> this miniPA.

Inline, we can give this miniPA.
Part III
History

v0.1.2: mini-PixelArts
v0.1.1: Bugfix with color
v0.1.0: Initial version