The string-diagrams package*
Draw string diagrams using TikZ

Paolo Brasolin
paolo.brasolin@gmail.com

v0.2.1 (2023/06/13)

Please note this is the major version zero, meant for initial development: *anything MAY change at any time.* The upside is that this is the best time to contribute! Of course you can also just keep the `sty` along with your code and not care at all.

1 Documentation

`/pgf/box`

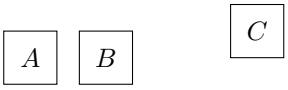
New: 2023-05-31
Updated: 2023-06-12

To draw boxes, you use this style on a node.

```
\begin{tikzpicture}
  \node[box] {A};
\end{tikzpicture}
```

You can draw multiple boxes using any of your standard Ti*k*Z positioning techniques. Don't forget to label the nodes so you can easily reference them.

```
\begin{tikzpicture}
  \node[box] (A) at (0,0) {A};
  \node[box, right of=A] (B) {B};
  \node[box] (C) at ($(B)+(2cm,1em)$) {C};
\end{tikzpicture}
```



*Thanks!

/pgf/box␣ports␣north /pgf/box ports north=⟨*integer*⟩
/pgf/box␣ports␣east /pgf/box ports east=⟨*integer*⟩
/pgf/box␣ports␣south /pgf/box ports south=⟨*integer*⟩
/pgf/box␣ports␣west /pgf/box ports west=⟨*integer*⟩

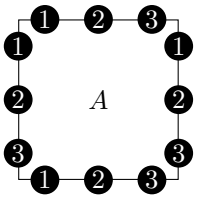
New: 2023-06-12 You can open up any number of ports on any side of a box using the appropriate key. Then, you can refer to the opened ports by their index.

```
\begin{tikzpicture}[
  marker/.style={circle, fill, inner sep=1pt, text=white},
]

\node[
  box,
  box ports north=3,
  box ports east=3,
  box ports south=3,
  box ports west=3,
  minimum width=6em,
  minimum height=6em,
] (A) {A};

\foreach \side in {north,east,south,west}
  \foreach \index in {1,...,3}
    \node[marker] at (A.\side.\index) {\index};

\end{tikzpicture}
```



/pgf/box␣ports /pgf/box ports=⟨*integer*⟩/⟨*integer*⟩/⟨*integer*⟩/⟨*integer*⟩

New: 2023-06-12 The `box ports` key is a shortcut to set the number of ports on all sides at once.

```
\begin{tikzpicture}[
  marker/.style={circle, fill, inner sep=1pt},
]

\node[box, box ports=1/2/3/4] (A) {A};

\foreach \side/\n in {north/1,east/2,south/3,west/4}
  \foreach \index in {1,...,\n}
    \node[marker] at (A.\side.\index) {};

\end{tikzpicture}
```

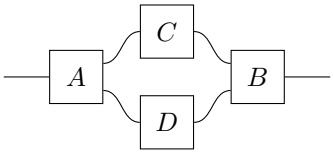
The same value can also be passed to the `box` key itself.

`\wires` `\wires[⟨TikZ keys⟩]{⟨connectivity⟩}{⟨loose ends⟩}`

New: 2023-05-31
Updated: 2023-06-13

To connect boxes, you can use the `\wires` macro. The first argument is Ti*k*Z styling for the wires; the second argument is a nested dicionary specifying the connectivity; the third argument is a list of the loose ends to draw. `boxes` have the following anchors: `west`, `west.0`, `west.1`, `east`, `east.0`, and `east.1`.

```
\begin{tikzpicture}[scale=0.6]
  \node[box=0/2/0/1] (A) at (-2, 0) {A};
  \node[box=0/1/0/2] (B) at (+2, 0) {B};
  \node[box=0/1/0/1] (C) at ( 0,+1) {C};
  \node[box=0/1/0/1] (D) at ( 0,-1) {D};
  \wires{
    A = { east.1 = C.west, east.2 = D.west },
    C = { east = B.west.1 },
    D = { east = B.west.2 },
  }{ A.west, B.east }
\end{tikzpicture}
```

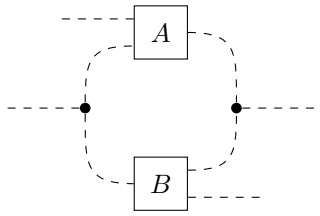


`/pgf/dot`

New: 2023-05-31

To split and join wires, you can use `dots` and their anchors `north`, `east`, `south`, and `west`. Remember to have fun with styling wires.

```
\begin{tikzpicture}
  \node[box=0/1/0/2] (A) at ( 0,+1) {A};
  \node[box=0/2/0/1] (B) at ( 0,-1) {B};
  \node[dot] (x) at (+1, 0) {};
  \node[dot] (y) at (-1, 0) {};
  \wires[looseness=1.5, dashed]{
    A = { east = x.north },
    B = { east.1 = x.south },
    y = { north = A.west.2, south = B.west },
  }{
    A.west.1, B.east.2, x.east, y.west
  }
\end{tikzpicture}
```



That's it. This is the package, for now.

2 Implementation

Open the DocStrip guards and set the internal namespace prefix (as per LaTeX3 DocStrip convention).

```
1 ⟨*package⟩
2 ⟨@@=stridi⟩
```

Load the essential support (expl3) "up-front".

```
3 \RequirePackage{expl3}[2023/05/11]
4 \RequirePackage{tikz}[2023/01/15]
```

Identify the package and give the over all version information.

```
\ProvidesExplPackage
  {string-diagrams}
  {2023/06/13}
  {0.2.1}
  {Draw string diagrams using TikZ}
```

/pgf/box␣ports␣north
/pgf/box␣ports␣east
/pgf/box␣ports␣south
/pgf/box␣ports␣west
/pgf/box␣ports

Define high level keys to configure the number of ports on each side.

```
\pgfkeys{
  /pgf/box~ports~north/.initial=1,
  /pgf/box~ports~east/.initial=1,
  /pgf/box~ports~south/.initial=1,
  /pgf/box~ports~west/.initial=1,
  /pgf/box~ports/.style~args={#1/#2/#3/#4}{
    /pgf/box~ports~north=#1,
    /pgf/box~ports~east=#2,
    /pgf/box~ports~south=#3,
    /pgf/box~ports~south=#4,
  },
}
```

(*End of definition for* `/pgf/box ports north` *and others. These functions are documented on page* 2.)

__stridi_intersect_hv_lines_through:NN

Calculates the intersection of two lines parallel to axes passing through given points on the plane.

`#1` : Point through which the vertical line passes
`#2` : Point through which the horizontal line passes

```
\cs_new:Nn \__stridi_intersect_hv_lines_through:NN {
  \pgfextractx { \pgf@xa } { #1 }
  \pgfextracty { \pgf@ya } { #2 }
  \pgfpoint { \pgf@xa } { \pgf@ya }
}
```

(*End of definition for* `\__stridi_intersect_hv_lines_through:NN`.)

__stridi_subdivide_segment:nNNNNN

Defines macros numbering equally spaced points on a segment.

`#1` : Base namespace
`#2` : Points count
`#3` : Point containing the x coordinate of the starting point
`#4` : Point containing the y coordinate of the starting point
`#5` : Point containing the x coordinate of the ending point
`#6` : Point containing the y coordinate of the ending point

```
\cs_new:Nn \__stridi_subdivide_segment:nNNNNN {
  \int_step_inline:nnnn { #2 } { -1 } { 1 } {
    \cs_if_exist:cTF
      { #1.##1 }
      { \prg_break: }
      { \prg_do_nothing: }
```

```
    \cs_new_nopar:cpn
      { #1.##1 }
      {
        \pgfmathdivide
          { 2 * ##1 - 1 }
          { 2 * #2 }
        \pgfpointlineattime
          { \pgfmathresult }
          { \__stridi_intersect_hv_lines_through:NN { #3 } { #4 } }
          { \__stridi_intersect_hv_lines_through:NN { #5 } { #6 } }
      }
  }
}
```

(*End of definition for* `\__stridi_subdivide_segment:nNNNNN`.)

`box` Define a rectangular shape with configurable ports.

```
\pgfdeclareshape{box}{

  % Inherit all the structure of rectangle
  \inheritsavedanchors[from=rectangle]
  \clist_map_inline:nn
    {
      north~west,  north, north~east,
            west, center,       east,
        mid~west,    mid,   mid~east,
       base~west,   base,  base~east,
      south~west,  south, south~east,
    }
    { \inheritanchor[from=rectangle]{#1} }
  \inheritanchorborder[from=rectangle]
  \inheritbackgroundpath[from=rectangle]

  % Dump port counts into saved macros
  \savedmacro\portsnorth
    {\pgfmathtruncatemacro\portsnorth{\pgfkeysvalueof{/pgf/box~ports~north}}}
  \savedmacro\portseast
    {\pgfmathtruncatemacro\portseast{\pgfkeysvalueof{/pgf/box~ports~east}}}
  \savedmacro\portssouth
    {\pgfmathtruncatemacro\portssouth{\pgfkeysvalueof{/pgf/box~ports~south}}}
  \savedmacro\portswest
    {\pgfmathtruncatemacro\portswest{\pgfkeysvalueof{/pgf/box~ports~west}}}

  % Add ports definitions to shape definition
  \expandafter\pgfutil@g@addto@macro\csname pgf@sh@s@box\endcsname{
    \__stridi_subdivide_segment:nNNNNN { pgf@anchor@box@north } { \portsnorth }
      { \southwest } { \northeast } { \northeast } { \northeast }
    \__stridi_subdivide_segment:nNNNNN { pgf@anchor@box@east } { \portseast }
      { \northeast } { \northeast } { \northeast } { \southwest }
    \__stridi_subdivide_segment:nNNNNN { pgf@anchor@box@south } { \portssouth }
      { \southwest } { \southwest } { \northeast } { \southwest }
    \__stridi_subdivide_segment:nNNNNN { pgf@anchor@box@west } { \portswest }
      { \southwest } { \northeast } { \southwest } { \southwest }
```

```
  }

}
```

(*End of definition for* box. *This function is documented on page* ??.)

/pgf/box Define style to draw boxes.

```
\ExplSyntaxOff
\tikzset{
  box/.default={0/0/0/0},
  box/.style args={#1}{
    shape=box,
    draw,
    inner sep=.5em,
    minimum width=2em,
    minimum height=2em,
    execute at begin node=$,
    execute at end node=$,
    /pgf/box ports=#1,
  },
}
\ExplSyntaxOn
```

(*End of definition for* /pgf/box. *This function is documented on page 1.*)

/pgf/dot Define style to draw dots.

```
\ExplSyntaxOff
\tikzset{
  dot/.style={
    shape=circle,
    fill,
    inner sep=0,
    minimum width=0.4em,
  },
}
\ExplSyntaxOn
```

(*End of definition for* /pgf/dot. *This function is documented on page 3.*)

__stridi_draw_bound_wires:nn Draws bound wires.

#1 : TikZ keys
#2 : dictionary of port labels

```
\cs_new:Nn \__stridi_draw_bound_wires:nn {
  \prop_set_from_keyval:Nn \l_tmpa_prop { #2 }
  \prop_map_inline:Nn \l_tmpa_prop
  {
    \prop_set_from_keyval:Nn \l_tmpb_prop { ##2 }
    \prop_map_inline:Nn \l_tmpb_prop
    {
      \regex_match_case:nn
```

```
      {
        { \. north } { \tl_gset:Nn \g_tmpa_tl { 90 } }
        { \. south } { \tl_gset:Nn \g_tmpa_tl { -90 } }
        { \. west } { \tl_gset:Nn \g_tmpa_tl { 180 } }
        { \. east } { \tl_gset:Nn \g_tmpa_tl { 0 } }
      } { ####2 }
      \regex_match_case:nn
      {
        { north } { \tl_gset:Nn \g_tmpb_tl { 90 } }
        { south } { \tl_gset:Nn \g_tmpb_tl { -90 } }
        { west } { \tl_gset:Nn \g_tmpb_tl { 180 } }
        { east } { \tl_gset:Nn \g_tmpb_tl { 0 } }
      } { ####1 }
      \draw [
        out={\tl_use:N \g_tmpb_tl},
        in={\tl_use:N \g_tmpa_tl},
        #1,
      ] (##1.####1) to (####2);
    }
  }
}
```

(*End of definition for* `\__stridi_draw_bound_wires:nn`.)

`\__stridi_draw_loose_wires:nn` Draws loose wires.

`#1` : TikZ keys
`#2` : list of port labels

```
\cs_new:Nn \__stridi_draw_loose_wires:nn {
  \clist_set:Nn \l_tmpa_clist { #2 }
  \clist_map_inline:Nn \l_tmpa_clist {
    \regex_match_case:nn
    {
      { \. north } { \draw[#1] (##1) -- +( 0,+1); } % TODO: cleaner solution?
      { \. south }
        {
          \draw[out=-90, in=0,#1] (##1)
            to ($(\pgf@picminx, \pgf@y)$);
        } % TODO: not sure why this works
      { \. west  } { \draw[#1] (##1) -- +(-1, 0); }
      { \. east  } { \draw[#1] (##1) -- +(+1, 0); }
    } { ##1 }
  }
}
```

(*End of definition for* `\__stridi_draw_loose_wires:nn`.)

`\wires` Define our main actor.

```
\NewDocumentCommand{\wires}{ O{} m m }
{
  \__stridi_draw_bound_wires:nn { #1 } { #2 }
  \__stridi_draw_loose_wires:nn { #1 } { #3 }
}
```

(*End of definition for* `\wires`. *This function is documented on page 3.*)

Close the DocStrip guards and call it a day.

```
160 ⟨/package⟩
```

Change History

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

R

S

T

W