Modern Beamer Presentations with the METROPOLIS package

Matthias Vogelgesang
matthias.vogelgesang@gmail.com

v0.x.x

Contents

1 Introduction

Beamer is an awesome way to make presentations with LaTeX, but its theme selection is surprisingly sparse. The stock themes share an aesthetic that is now overused and can be a little cluttered, and the few distinctive custom themes available are often specialized for a particular corporate or institutional brand.

The goal of METROPOLIS is to provide a simple, modern Beamer theme suitable for anyone to use. It tries to minimize noise and maximize space for content; the only visual flourish it offers is an (optional) progress bar added to each slide or to the section slides.

By default, METROPOLIS uses Fira Sans, a gorgeous typeface commissioned by Mozilla and designed by Carrois. For best results, you will need the Fira typeface installed and use XeLaTeX to typeset your slides. However, METROPOLIS can also be used other typefaces and LaTeX build systems.

METROPOLIS's codebase is maintained on GitHub. If you have issues, find mistakes in the manual or want to help make the theme even better, please get in touch there. The full list of contributors already contains over a dozen names!

2 Getting Started

2.1 Installing from GitHub

Installing METROPOLIS, like any Beamer theme, involves four easy steps:

Download the source with a `git clone` of the METROPOLIS repository or as a zip archive of the latest development version.

Compile the style files by running `make sty` inside the downloaded directory. (Or run LaTeX directly on `source/metropolistheme.ins`.)

Move the resulting `*.sty` files to the folder containing your presentation. To use METROPOLIS with many presentations, run `make install` or move the `*.sty` files to a folder in your TeX path instead.

Use the theme for your presentation by declaring `\usetheme{m}` in the preamble of your Beamer document.

METROPOLIS uses the Make build system to offer the following installation options for advanced users:

`make sty` builds the theme style files.

`make doc` builds this documentation manual.

`make demo` builds a demo presentation to test the features of METROPOLIS.

`make all` builds the theme, manual, and demo presentation.

`make clean` removes the files generated by `make all`.

`make install` installs the theme into your local texmf folder.

`make uninstall` removes the theme from your local texmf folder.

`make ctan` creates a package for CTAN distribution.

2.2 A Minimal Example

The following code shows a minimal example of a Beamer presentation using METROPOLIS.

```
\documentclass{beamer}
\usetheme{m}                     % Use metropolis theme
\title{A minimal example}
\date{\today}
\author{Matthias Vogelgesang}
\institute{Centre for Modern Beamer Themes}
\begin{document}
  \maketitle
  \section{First Section}
  \begin{frame}{First Frame}
    Hello, world!
  \end{frame}
\end{document}
```

2.3 Dependencies

- XeLaTeX
- Fira Sans and Mono font
- TikZ

Depending on the Linux distribution, the packaged name of `Fira Sans` might be `Fira Sans OT` instead of `Fira Sans`. In that case, you may have to edit `beamerfontthememetropolis.dtx`. You may also need to install Fira Sans; see the `contrib/` directory for more. Users of Debian or Ubuntu can also install this .deb package containing the theme files as well as the Fira Sans font files.

2.4 Pandoc

To use this theme with Pandoc-based presentations, you can run the following command

```
$ pandoc -t beamer --latex-engine=xelatex -V theme:m -o
   output.pdf input.md
```

3 Customization

3.1 Package options

The theme provides a number of options. The options use a key=value interface. So every option is controlled by a key its value. To use an option you can either provide a comma separated list of options when invoking mtheme in the preamble of the presentation.

```
\usetheme[<key=value list>]{m}
```

Or you can set them at any time with the `\metroset` macro.

```
\metroset[<key=value list>]
```

To set an option on a specific sub-package only you have to add the corresponding prefix (inner, outer, color), e.g.

```
\metroset[inner/block=fill]
```

The list of options is structured as shown in the following example.

`key` *list of possible values* . . . . default value

A short description of the option.

Although the options are grouped into the corresponding packages every option can and in most cases should be set on the main theme directly. If an option is listed in multiple sub-packages, setting it on the main theme will set the option on every sub-package accordingly.

3.1.1 Main theme

`everytitleformat` *regular, lowercase, uppercase* . . . . lowercase

Shortcut option to change the case style of all titles together.

`plaintitleformat` *regular, lowercase, uppercase* . . . . lowercase

Control the case style of the plain title.

3.1.2 Inner theme

`block` *transparent, fill* . . . . transparent

This option controls the block background. It can either be filled with a light grey or be transparent.

`sectionpage` *none, progressbar* . . . . progressbar

Adds a thin progress bar similar to the section progress bar underneath each frame title.

`titleformat` *regular, lowercase, uppercase* . . . . lowercase

Control the case style of the title.

`sectiontitleformat` *regular, lowercase, uppercase* . . . . lowercase

Control the case style of the section title.

3.1.3 Outer theme

`numbering` *none, counter, fraction* . . . . counter

In the bottom right corner of each frame the current frame number is displayed. This can be disabled or the total framenumber can be added additionally.

`progressbar` *none, head, frametitle, foot* . . . . none

Adds a progress bar to the top of each frame (`head`), the bottom of each frame (`foot`), or directly below each frame title (`frametitle`).

`frametitleformat` *regular, lowercase, uppercase* . . . . lowercase

Control the case style of the frame title.

3.1.4 Color theme

`block` *transparent, fill* . . . . transparent

This option controls the block background. It can either be filled with a light grey or be transparent.

`background` *dark, light* . . . . light

This option defines whether the background shall be dark and the foreground be light or vice versa.

3.2 Color Customization

The included METROPOLIS color theme is used by default, but its colors can be easily changed to suit your tastes. All of the theme's styles are defined in terms of three beamer colors:

- `normal text` (dark fg, light bg)
- `alerted text` (colored fg, should be visible against dark or light)
- `example text` (colored fg, should be visible against dark or light)

An easy way to customize the theme is to redefine these colors using

```
\setbeamercolor{ ... }{ fg= ... , bg= ... }
```

in your preamble. For greater customization, you can redefine any of the other stock beamer colors. In addition to the stock colors the theme defines a number of METROPOLIS specific colors, which can also be redefined to your liking.

```
\setbeamercolor{progress bar}{ ... }
```

```
\setbeamercolor{title separator}{ ... }
\setbeamercolor{progress bar in head/foot}{ ... }
\setbeamercolor{progress bar in section page}{ ... }
```

3.3 Commands

The `\plain{title=[]}{<body>}` command sets a slide in plain dark colors which can be useful to focus attention on a single sentence or image.

3.4 Paul Tol's colors: a `pgfplots` theme

A good presentation uses colors that are

- distinct from each other as much as possible, and
- distinct from black and white,
- under many different lighting and display environments, and
- to color-blind viewers,
- all while matching well together.

In a technical note for SRON, Paul Tol proposed a palette of colors satisfying these constraints. The sub-package `pgfplotsthemetol` defines palettes for `pgfplots` charts based on Tol's work. Use the `mlineplot` key to plot line data and `mbarplot` or horizontal `mbarplot` to plot bar charts.

4 Known Issues

The `\plain` command does not work if you override the METROPOLIS color theme with the default beamer color theme `fly`.

5 License

The theme itself is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License. This means that if you change the theme and re-distribute it, you must retain the copyright notice header and license it under the same CC-BY-SA license. This does not affect the presentation that you create with the theme.

6 Implementation

6.1 METROPOLIS main theme

The primary job of this package is to load the component sub-packages of the METROPOLIS theme and route the theme options accordingly. It also provides some custom commands and environments for the user.

Load the required packages.

```
\RequirePackage{etoolbox}
\RequirePackage{pgfopts}
\RequirePackage{ifxetex}
\RequirePackage{ifluatex}
```

6.1.1 Options

\metroset First of all we define a macro for the user to set options.

```
\newcommand{\metroset}[1]{\pgfkeys{/metropolis/.cd,#1}}
```

Then we need to pass the unknown options to the sub-packages.

```
\pgfkeys{/metropolis/.cd,
  .search also={
    /metropolis/inner,
    /metropolis/outer,
    /metropolis/color,
  },
```

We have to forwarded keys that affect multiple sub-packages manually.

```
  block/.code=\pgfkeysalso{
    inner/block=#1,
    color/block=#1,
  },
}
```

`plaintitleformat` Control the case style of the plain title

```
\pgfkeys{
  /metropolis/plaintitleformat/.cd,
    .is choice,
    regular/.code=\renewcommand{\@metropolis@plaintitleformat}{#1},
    lowercase/.code={%
      \renewcommand{\@metropolis@plaintitleformat}{\MakeLowercase{#1}}
    },
    uppercase/.code={%
      \renewcommand{\@metropolis@plaintitleformat}{\MakeUppercase{#1}}
    },
}
```

`everytitleformat` Control the case style of the every title

```
\pgfkeys{
  /metropolis/everytitleformat/.code=\pgfkeysalso{
      inner/titleformat=#1,
      inner/sectiontitleformat=#1,
      outer/frametitleformat=#1,
      plaintitleformat=#1,
    }
}
```

For backwards compatibility with earlier betas of the theme, we implement deprecated option names as aliases to the corresponding `key=value` options.

```
\pgfkeys{/metropolis/.cd,
  usetitleprogressbar/.code=\pgfkeysalso{outer/progressbar=frametitle},
  noslidenumbers/.code=\pgfkeysalso{outer/numbering=none},
  usetotalslideindicator/.code=\pgfkeysalso{outer/numbering=fraction},
  nosectionslide/.code=\pgfkeysalso{inner/sectionpage=none},
  darkcolors/.code=\pgfkeysalso{color/background=dark},
  blockbg/.code=\pgfkeysalso{color/block=fill, inner/block=fill},
}
```

Set default values for options.

```
\newcommand{\@metropolis@setdefaults}{
  \pgfkeys{/metropolis/.cd,
    plaintitleformat=lowercase,
  }
}
```

6.1.2 Component sub-packages

Having processed the options, we can now load the component sub-packages of the theme.

```
\useinnertheme{metropolis}
\useoutertheme{metropolis}
\usecolortheme{metropolis}
```

The `fira` font theme, which depends on `fontspec`, is only loaded if the document is being processed by XeLaTeX or LuaLaTeX.

```
\ifboolexpr{bool {xetex} or bool {luatex}}{
  \usefonttheme{metropolis}
}{
  \PackageWarning{beamerthemem}{%
    You need to compile with XeLaTeX or LuaLaTeX to use the Fira fonts.
  }
}
```

The `tol` theme for `pgfplots` is only loaded if `pgfplots` is used.

```
\AtEndPreamble{%
  \@ifpackageloaded{pgfplots}{%
    \RequirePackage{pgfplotsthemetol}
  }{}
}
```

6.1.3 Custom commands

We define custom commands in this package as their proper usage may depend on multiple sub-packages.

`metropolis@plaintitleformat` Define a hook to change the case format of the plain title.

```
\def\@metropolis@plaintitleformat#1{#1}
```

`\plain` Creates a plain frame with dark background, suitable for displaying images or a few words.

```
\newcommand{\plain}[2][]{%
  \begingroup
    \setbeamercolor{background canvas}{
      use=palette primary,
      parent=palette primary
    }
    \begin{frame}[c]{#1}
      \begin{center}
        \usebeamercolor[fg]{palette primary}
        \usebeamerfont{section title}
        \@metropolis@plaintitleformat{#2}
      \end{center}
    \end{frame}
  \endgroup
}
```

`\mreducelistspacing`

```
\newcommand{\mreducelistspacing}{\vspace{-\topsep}}
```

Process package options

```
\@metropolis@setdefaults
\ProcessPgfOptions{/metropolis}
```

6.2 METROPOLIS inner theme

A `beamer` inner theme dictates the style of the frame elements traditionally set in the “body” of each slide. These include:

- title, part, and section pages;
- itemize, enumerate, and description environments;
- block environments including theorems and proofs;

- figures and tables; and
- footnotes and plain text.

Load required packages.

```
\RequirePackage{etoolbox}
\RequirePackage{calc}
\RequirePackage{pgfopts}
\RequirePackage{tikz}
```

6.2.1 Options

block This option controls the block style.

```
\pgfkeys{
  /metropolis/inner/block/.cd,
    .is choice,
    transparent/.code=\setlength{\@metropolis@blockskip}{0ex},
    fill/.code=\setlength{\@metropolis@blockskip}{1ex},
}
```

titleformat Control the case style of the title

```
\pgfkeys{
  /metropolis/inner/titleformat/.cd,
    .is choice,
    regular/.code=\renewcommand{\@metropolis@titleformat}{},
    lowercase/.code={%
      \renewcommand{\@metropolis@titleformat}{\MakeLowercase}
    },
    uppercase/.code={%
      \renewcommand{\@metropolis@titleformat}{\MakeUppercase}
    },
}
```

sectiontitleformat Control the case style of the section title

```
\pgfkeys{
  /metropolis/inner/sectiontitleformat/.cd,
    .is choice,
    regular/.code=\renewcommand{\@metropolis@sectiontitleformat}{},
    lowercase/.code={%
      \renewcommand{\@metropolis@sectiontitleformat}{\MakeLowercase}
    },
    uppercase/.code={%
      \renewcommand{\@metropolis@sectiontitleformat}{\MakeUppercase}
    },
}
```

`sectionpage`

The `sectionpage` option defines the behaviour of the sectionpage.

```
\pgfkeys{
  /metropolis/inner/sectionpage/.cd,
    .is choice,
    none/.code=\@metropolis@sectionpage@none,
    progressbar/.code=\@metropolis@sectionpage@progressbar,
}
```

`etropolis@inner@setdefaults`

Set default values for inner theme options.

```
\newcommand{\@metropolis@inner@setdefaults}{
  \pgfkeys{/metropolis/inner/.cd,
    sectionpage=progressbar,
    block=transparent,
    titleformat=lowercase,
    sectiontitleformat=lowercase,
  }
}
```

6.2.2 Title page

`\@metropolis@titleformat`

Define hooks to change the case format of the titles.

```
\def\@metropolis@titleformat#1{#1}
\def\@metropolis@sectiontitleformat#1{#1}
```

To make the `\MakeLowercase` and `\MakeUppercase` macros work in the sectiontitle we have to patch `\sectionentry` and `\beamer@section`. This solution was suggested by Enrico Gregorio in an answer to this StackExchange question.

```
\patchcmd{\sectionentry}
  {\def\insertsectionhead{#2}}
  {\def\insertsectionhead{\@metropolis@sectiontitleformat{#2}}}
  {}
  {\PackageError{beamerinnerthememetropolis}{Patching section title failed.}}
\patchcmd{\beamer@section}
  {\def\insertsectionhead{\hyperlink{Navigation\the\c@page}{#1}}}
  {\def\insertsectionhead{\hyperlink{Navigation\the\c@page}{\@metropolis@sectiontit
  {}
  {\PackageError{beamerinnerthememetropolis}{Patching section title failed.}}
```

`title graphic` Set the title graphic in a zero-height box, so it doesn't change the position of other elements.

```
\setbeamertemplate{title graphic}{
  \vbox to 0pt {
    \vspace*{2em}
    \inserttitlegraphic%
  }%
  \nointerlineskip%
}
```

`title` Set the title on the title page.

```
\setbeamertemplate{title}{
  \raggedright%
  \linespread{1.0}%
  \@metropolis@titleformat{\inserttitle}%
  \par%
  \vspace*{0.5em}
}
```

`subtitle` Set the subtitle on the title page.

```
\setbeamertemplate{subtitle}{
  \insertsubtitle%
  \par%
  \vspace*{0.5em}
}
```

title separator Template to set the title graphic in a zero-height box. (It won't change the position of other elements.)

```
\setbeamertemplate{title separator}{
  \begin{tikzpicture}
    \draw[fg] (0, 0) -- (\textwidth, 0);
  \end{tikzpicture}%
  \par%
}
```

author Set the author on the title page.

```
\setbeamertemplate{author}{
  \vspace*{2em}
  \insertauthor%
  \par%
  \vspace*{0.25em}
}
```

date Set the date on the title page.

```
\setbeamertemplate{date}{
  \insertdate%
  \par%
}
```

institute Set the institute on the title page.

```
\setbeamertemplate{institute}{
  \vspace*{3mm}
  \insertinstitute%
  \par%
}
```

title page Template for the title page. Each element is only typset if it is defined by the user. If \subtitle is empty, for example, it won't leave a blank space on the title slide.

```
\setbeamertemplate{title page}{
  \begin{minipage}[b][\paperheight]{\textwidth}
    \ifx\inserttitlegraphic\@empty\else\usebeamertemplate*{title graphic}\fi
    \vfill%
    \ifx\inserttitle\@empty\else\usebeamertemplate*{title}\fi
    \ifx\insertsubtitle\@empty\else\usebeamertemplate*{subtitle}\fi
    \usebeamertemplate*{title separator}
```

Beamer's definition of `\insertauthor` is always nonempty, so we have to test another macro initialized by `\author{...}` to see if the user has defined an author. This solution was suggested by Enrico Gregorio in an answer to this Stack Exchange question.

```
    \ifx\beamer@shortauthor\@empty\else\usebeamertemplate*{author}\fi
    \ifx\insertdate\@empty\else\usebeamertemplate*{date}\fi
    \ifx\insertinstitute\@empty\else\usebeamertemplate*{institute}\fi
    \vfill
    \vspace*{1mm}
  \end{minipage}
}
```

Normal people should use `\maketitle` or `\titlepage` instead of using the `title page` beamer template directly. Beamer already defines these macros, but we patch them here to make the title page `[plain]` by default, remove `\@thanks`, and ensure the title frame number doesn't count.

`\maketitle`
`\titlepage`

Inserts the title frame, or causes the current frame to use the `title page` template.

```
\def\maketitle{%
  \ifbeamer@inframe
    \titlepage
  \else
    \frame[plain]{\titlepage}
  \fi
}
\def\titlepage{%
  \usebeamertemplate{title page}
}
```

6.2.3 Section page

section page Template for the section title slide at the beginning of each section.

```
\newcommand{\@metropolis@sectionpage@none}{
  \AtBeginSection{
    % intenionally empty
  }
}
\defbeamertemplate{section page}{progressbar}{
  \centering
  \begin{minipage}{22em}
    \usebeamercolor[fg]{section title}
    \usebeamerfont{section title}
    \insertsectionhead\\[-1ex]
    \usebeamertemplate*{progress bar in section page}
  \end{minipage}
  \par
}
\newcommand{\@metropolis@sectionpage@progressbar}{
  \setbeamertemplate{section page}[progressbar]
  \AtBeginSection{
    \ifbeamer@inframe
      \sectionpage
    \else
      \frame[plain,c]{\sectionpage}
    \fi
  }
}
```

rogress bar in section page Template for the progress bar displayed by default on the section page. This code is duplicated in large part in the outer theme's template `progress bar in head-/foot`.

```
\newlength{\metropolis@progressonsectionpage}
\setbeamertemplate{progress bar in section page}{
  \setlength{\metropolis@progressonsectionpage}{%
    \textwidth * \ratio{\insertframenumber pt}{\inserttotalframenumber pt}%
  }%
  \begin{tikzpicture}
```

```
    \draw[bg, fill=bg] (0,0) rectangle (\textwidth, 0.4pt);
    \draw[fg, fill=fg] (0,0) rectangle (\metropolis@progressonsectionpage, 0.4pt);
  \end{tikzpicture}%
}
```

The above code assumes that `\insertframenumber` is less than or equal to `\inserttotalframenumber`. However, this is not true on the first compile; in the absence of an `.aux` file, `\inserttotalframenumber` defaults to 1. This behaviour could cause fatal errors for long presentations, as `\metropolis@progressonsectionpag` would exceed TEX's maximum length (16383.99999pt, roughly 5.75 metres or 18.9 feet). To avoid this, we increase the default value for `\inserttotalframenumber`; presentations with over 4000 slides will still break on first compile, but users in that situation likely have deeper problems to solve.

```
\def\inserttotalframenumber{100}
```

6.2.4 Block environments

Regular block environment

```
\newlength{\@metropolis@blockskip}
\setbeamertemplate{block begin}{%
  \vspace*{1ex}
  \begin{beamercolorbox}[%
    ht=2.4ex,
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip]{block title}
      \usebeamerfont*{block title}\insertblocktitle%
  \end{beamercolorbox}%
  \vspace*{-1pt}
  \usebeamerfont{block body}%
  \begin{beamercolorbox}[%
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip,
    vmode]{block body}%
}
\setbeamertemplate{block end}{%
```

```
  \end{beamercolorbox}
  \vspace*{0.2ex}
}
```

Alerted block environment

```
\setbeamertemplate{block alerted begin}{%
  \vspace*{1ex}
  \begin{beamercolorbox}[%
    ht=2.4ex,
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip]{block title alerted}
      \usebeamerfont*{block title alerted}\insertblocktitle%
  \end{beamercolorbox}%
  \vspace*{-1pt}
  \usebeamerfont{block body alerted}%
  \begin{beamercolorbox}[%
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip,
    vmode]{block body}%
}
\setbeamertemplate{block alerted end}{%
  \end{beamercolorbox}
  \vspace*{0.2ex}
}
```

Example block environment

```
\setbeamertemplate{block example begin}{%
  \vspace*{1ex}
  \begin{beamercolorbox}[%
    ht=2.4ex,
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip]{block title example}
      \usebeamerfont*{block title example}\insertblocktitle%
  \end{beamercolorbox}%
  \vspace*{-1pt}
  \usebeamerfont{block body example}%
  \begin{beamercolorbox}[%
    dp=1ex,
    leftskip=\@metropolis@blockskip,
    rightskip=\@metropolis@blockskip,
    vmode]{block body}%
}
\setbeamertemplate{block example end}{%
  \end{beamercolorbox}
  \vspace*{0.2ex}
}
```

6.2.5 Lists and floats

```
\setbeamertemplate{itemize items}{\textbullet}
\setbeamertemplate{caption label separator}{: }
\setbeamertemplate{caption}[numbered]
```

6.2.6 Footnotes

```
\setbeamertemplate{footnote}{%
  \parindent 0em\noindent%
  \raggedright
  \usebeamercolor{footnote}\hbox to 0.8em{\hfil\insertfootnotemark}\insertfootnotete
}
```

6.2.7 Text and spacing settings

```
\setlength{\parskip}{0.5em}
\linespread{1.15}
```

By default, Beamer frames offer the `c` option to *almost* vertically center the text, but the placement is a little too high. To fix this, we redefine the `c` option to equalize `\beamer@frametopskip` and `\beamer@framebottomskip`. This solution was suggested by Enrico Gregorio in an answer to this Stack Exchange question.

```
\define@key{beamerframe}{c}[true]{% centered
  \beamer@frametopskip=0pt plus 1fill\relax%
  \beamer@framebottomskip=0pt plus 1fill\relax%
  \beamer@frametopskipautobreak=0pt plus .4\paperheight\relax%
  \beamer@framebottomskipautobreak=0pt plus .6\paperheight\relax%
  \def\beamer@initfirstlineunskip{}%
}
```

Process package options

```
\@metropolis@inner@setdefaults
\ProcessPgfPackageOptions{/metropolis/inner}
```

6.3 METROPOLIS outer theme

A `beamer` outer theme dictates the style of the frame elements traditionally set outside the body of each slide: the head, footline, and frame title.

Load required packages.

```
\RequirePackage{etoolbox}
\RequirePackage{calc}
\RequirePackage{pgfopts}
```

6.3.1 Options

`numbering` This option controls the page numbering.

```
\pgfkeys{
  /metropolis/outer/numbering/.cd,
    .is choice,
    none/.code=\setbeamertemplate{frame numbering}[none],
    counter/.code=\setbeamertemplate{frame numbering}[counter],
    fraction/.code=\setbeamertemplate{frame numbering}[fraction],
}
```

`progressbar` This option controls the progressbar.

```
\pgfkeys{
  /metropolis/outer/progressbar/.cd,
    .is choice,
    none/.code={%
      \setbeamertemplate{headline}[plain]
      \setbeamertemplate{frametitle}[plain]
      \setbeamertemplate{footline}[plain]
    },
    head/.code={\pgfkeys{/metropolis/outer/progressbar=none}
      \addtobeamertemplate{headline}{}{\usebeamertemplate*{progress bar in head/foot}}
    },
    frametitle/.code={\pgfkeys{/metropolis/outer/progressbar=none}
      \addtobeamertemplate{frametitle}{}{\usebeamertemplate*{progress bar in head/foot}}
    },
    foot/.code={\pgfkeys{/metropolis/outer/progressbar=none}
      \addtobeamertemplate{footline}{}{\usebeamertemplate*{progress bar in head/foot}}
    },
}
```

`frametitleformat` Control the case style of the frame title

```
\pgfkeys{
  /metropolis/outer/frametitleformat/.cd,
    .is choice,
    regular/.code={%
       \renewcommand{\@metropolis@frametitleformat}{}%
       \renewcommand{\@metropolis@frametitlestrut}{%
           \rule{0pt}{\heightof{ABCDEFGHIJKLMNOPQRSTUVWXYZ}}
       }
       },
    lowercase/.code={%
       \renewcommand{\@metropolis@frametitleformat}{\MakeLowercase}%
       \renewcommand{\@metropolis@frametitlestrut}{%
           \rule{0pt}{\heightof{abcdefghijklmnopqrstuvwxyz}}
       }
       },
    uppercase/.code={%
       \renewcommand{\@metropolis@frametitleformat}{\MakeUppercase}%
       \renewcommand{\@metropolis@frametitlestrut}{%
           \rule{0pt}{\heightof{ABCDEFGHIJKLMNOPQRSTUVWXYZ}}
       }
       },
}
```

etropolis@outer@setdefaults Set default values for outer theme options.

```
\newcommand{\@metropolis@outer@setdefaults}{
  \pgfkeys{/metropolis/outer/.cd,
    numbering=counter,
    progressbar=none,
    frametitleformat=lowercase,
  }
}
```

6.3.2 Head and footline

All good `beamer` presentations should already remove the navigation symbols, but METROPOLIS removes them automatically (just in case).

```
\setbeamertemplate{navigation symbols}{}
```

Templates for the frame number. Can be omitted, shown or displayed as a fraction of the total frames.

```
\defbeamertemplate{frame numbering}{none}{}
\defbeamertemplate{frame numbering}{counter}{\insertframenumber}
\defbeamertemplate{frame numbering}{fraction}{
  \insertframenumber/\inserttotalframenumber
}
```

```
\defbeamertemplate{headline}{plain}{}
\defbeamertemplate{footline}{plain}{%
  \begin{beamercolorbox}[wd=\textwidth, sep=3ex]{footline}%
    \hfill%
    \usebeamerfont{page number in head/foot}%
    \usebeamertemplate*{frame numbering}
  \end{beamercolorbox}%
}
```

6.3.3 Frametitle

metropolis@frametitleformat Define a hook to change the case format of the frame title.

```
\def\@metropolis@frametitleformat#1{#1}
```

To make the \MakeLowercase and \MakeUppercase macros work in the frame title we have to patch \beamer@@frametitle. This solution was suggested by Enrico Gregorio in an answer to this StackExchange question.

```
\patchcmd{\beamer@@frametitle}
  {\beamer@ifempty{#2}{}{%
      \gdef\insertframetitle{{#2\ifnum\beamer@autobreakcount>0\relax{}\space\usebea
tinuation}\fi}}%
    \gdef\beamer@frametitle{#2}%
    \gdef\beamer@shortframetitle{#1}%
    }}
  {\beamer@ifempty{#2}{}{%
      \gdef\insertframetitle{{\@metropolis@frametitleformat{#2}\ifnum\beamer@autobr
tinuation}\fi}}%
    \gdef\beamer@frametitle{#2}%
    \gdef\beamer@shortframetitle{#1}%
    }}
  {}
  {\PackageError{beamerouterthememetropolis}{Patching frame title failed.}}
```

frametitle Templates for the frame title, which is optionally underlined with a progress bar.

```
\newlength{\@metropolis@frametitlestrut}
\defbeamertemplate{frametitle}{plain}{%
  \nointerlineskip%
  \begin{beamercolorbox}[%
      wd=\paperwidth,%
      sep=1.5ex,%
    ]{frametitle}%
  \@metropolis@frametitlestrut\insertframetitle\@metropolis@frametitlestrut%
  \end{beamercolorbox}%
}
```

progress bar in head/foot Template for the progress bar optionally displayed below the frame title on each page. Much of this code is duplicated in the inner theme's template progress bar in section page.

```
\newlength{\metropolis@progressinheadfoot}
\setbeamertemplate{progress bar in head/foot}{
  \nointerlineskip
  \setlength{\metropolis@progressinheadfoot}{%
```

```
    \paperwidth * \ratio{\insertframenumber pt}{\inserttotalframenumber pt}%
  }%
  \begin{beamercolorbox}[wd=\paperwidth]{progress bar in head/foot}
    \begin{tikzpicture}
      \draw[bg, fill=bg] (0,0) rectangle (\paperwidth, 0.4pt);
      \draw[fg, fill=fg] (0,0) rectangle (\metropolis@progressinheadfoot, 0.4pt);
    \end{tikzpicture}%
  \end{beamercolorbox}
}
```

Process package options

```
\@metropolis@outer@setdefaults
\ProcessPgfPackageOptions{/metropolis/outer}
```

6.4 Fira font theme

Font Definitions

```
\RequirePackage[no-math]{fontspec}
\defaultfontfeatures{Mapping=tex-text}
\setsansfont[BoldItalicFont={Fira Sans Italic},%
             ItalicFont={Fira Sans Light Italic},%
             BoldFont={Fira Sans}]{Fira Sans Light}
\setmonofont{Fira Mono}
\newfontfamily\ExtraLight{Fira Sans ExtraLight}
\newfontfamily\Light{Fira Sans Light}
\newfontfamily\Book{Fira Sans}
\newfontfamily\Medium{Fira Sans Medium}
\AtBeginEnvironment{tabular}{%
    \setsansfont[BoldFont={Fira Sans},%
                 Numbers={Monospaced}]{Fira Sans Light}%
    }
```

Font Assignment

```
\setbeamerfont{title}{family=\Book, size=\Large, shape=\scshape}
\setbeamerfont{author}{family=\ExtraLight, size=\small}
\setbeamerfont{date}{family=\ExtraLight, size=\small}
\setbeamerfont{section title}{family=\Book, size=\Large, shape=\scshape}
```

```
\setbeamerfont{block title}{family=\Book, size=\normalsize}
\setbeamerfont{block title alerted}{family=\Book,size=\normalsize}
\setbeamerfont{subtitle}{family=\Light, size=\fontsize{12}{14}}
\setbeamerfont{frametitle}{family=\Book, size=\large, shape=\scshape}
\setbeamerfont{caption}{size=\small}
\setbeamerfont{caption name}{family=\Book}
\setbeamerfont{description item}{family=\Book}
\setbeamerfont{page number in head/foot}{size=\scriptsize}
```

Bibliograpy

```
\setbeamerfont{bibliography entry author}{family=\Light, size=\normalsize}
\setbeamerfont{bibliography entry title}{family=\Book, size=\normalsize}
\setbeamerfont{bibliography entry location}{family=\Light, size=\normalsize}
\setbeamerfont{bibliography entry note}{family=\Light, size=\small}
\linespread{1.15}
```

6.5 metropolis color theme

Load required packages.

```
\RequirePackage{pgfopts}
```

6.5.1 Options

block This option controls whether the blocks are filled or transparent.

```
\pgfkeys{
  /metropolis/color/block/.cd,
    .is choice,
    transparent/.code=\@metropolis@block@transparent,
    fill/.code=\@metropolis@block@fill,
}
```

colors Defines whether the background shall be dark and the foreground be light or vice versa

```
\pgfkeys{
  /metropolis/color/background/.cd,
    .is choice,
    dark/.code=\@metropolis@colors@dark,
    light/.code=\@metropolis@colors@light,
}
```

etropolis@color@setdefaults Set default values for color theme options.

```
\newcommand{\@metropolis@color@setdefaults}{
  \pgfkeys{/metropolis/color/.cd,
    background=light,
    block=transparent,
  }
}
```

6.5.2 Base colors

```
\definecolor{mDarkBrown}{HTML}{604c38}
\definecolor{mDarkTeal}{HTML}{23373b}
\definecolor{mLightBrown}{HTML}{EB811B}
\definecolor{mLightGreen}{HTML}{14B03D}
```

6.5.3 Base styles

All colors in the METROPOLIS theme are derived from the definitions of `normal text`, `alerted text`, and `example text`.

```
\newcommand{\@metropolis@colors@dark}{
  \setbeamercolor{normal text}{%
    fg=black!2,
    bg=mDarkTeal
  }
}
\newcommand{\@metropolis@colors@light}{
  \setbeamercolor{normal text}{%
    fg=mDarkTeal,
    bg=black!2
  }
}
\setbeamercolor{alerted text}{%
  fg=mLightBrown
}
```

```
\setbeamercolor{example text}{%
  fg=mLightGreen
}
```

6.5.4 Derived colors

The titles and structural elements (e.g. `itemize` bullets) are set in the same color as `normal text`. This would ideally done by setting `normal text` as a parent style, which we do to set `titlelike`, but this doesn't work for `structure` as its foreground is set explicitly in `beamercolorthemedefault.sty`.

```
\setbeamercolor{titlelike}{use=normal text, parent=normal text}
\setbeamercolor{author}{use=normal text, parent=normal text}
\setbeamercolor{institute}{use=normal text, parent=normal text}
\setbeamercolor{structure}{use=normal text, fg=normal text.fg}
```

The "primary" palette should be used for the most important navigational elements, and possibly of other elements. The METROPOLIS theme uses it for frame titles and slides.

```
\setbeamercolor{palette primary}{%
  use=normal text,
  fg=normal text.bg,
  bg=normal text.fg
}
\setbeamercolor{frametitle}{%
  use=palette primary,
  parent=palette primary
}
```

The METROPOLIS inner or outer themes optionally display progress bars in various locations. Their color is set by `progress bar` but the two different kinds can be customized separately. The horizontal rule on the title page is also set based on the progress bar color and can be customized with `title separator`.

```
\setbeamercolor{progress bar}{%
  use=alerted text,
  fg=alerted text.fg,
  bg=normal text.bg!50!normal text.fg
}
```

```
\setbeamercolor{title separator}{
  use=progress bar,
  parent=progress bar
}
\setbeamercolor{progress bar in head/foot}{%
  use=progress bar,
  parent=progress bar
}
\setbeamercolor{progress bar in section page}{
  use=progress bar,
  parent=progress bar
}
```

Blocks

```
\newcommand{\@metropolis@block@transparent}{
  \setbeamercolor{block title}{use=normal text, parent=normal text}
}
\newcommand{\@metropolis@block@fill}{
  \setbeamercolor{block title}{%
    use=normal text,
    fg=normal text.fg,
    bg=normal text.bg!80!fg
  }
}
\setbeamercolor{block title alerted}{%
    use={block title, alerted text},
    bg=block title.bg,
    fg=alerted text.fg
}
\setbeamercolor{block title example}{%
    use={block title, example text},
    bg=block title.bg,
    fg=example text.fg
}
\setbeamercolor{block body alerted}{use=block body, parent=block body}
\setbeamercolor{block body example}{use=block body, parent=block body}
\setbeamercolor{block body}{
  use={block title, normal text},
  bg=block title.bg!50!normal text.bg
}
```

Footnotes

```
\setbeamercolor{footnote}{fg=normal text.fg!90}
\setbeamercolor{footnote mark}{fg=.}
```

Process package options

```
\@metropolis@color@setdefaults
\ProcessPgfPackageOptions{/metropolis/color}
```

```
\mode<all>
```

6.6 Tol `pgfplots` theme

Paul Tol's 12-color palette[1] is as follows:

```
\definecolor{TolDarkPurple}{HTML}{332288}
\definecolor{TolDarkBlue}{HTML}{6699CC}
\definecolor{TolLightBlue}{HTML}{88CCEE}
\definecolor{TolLightGreen}{HTML}{44AA99}
\definecolor{TolDarkGreen}{HTML}{117733}
\definecolor{TolDarkBrown}{HTML}{999933}
\definecolor{TolLightBrown}{HTML}{DDCC77}
\definecolor{TolDarkRed}{HTML}{661100}
\definecolor{TolLightRed}{HTML}{CC6677}
\definecolor{TolLightPink}{HTML}{AA4466}
\definecolor{TolDarkPink}{HTML}{882255}
\definecolor{TolLightPurple}{HTML}{AA4499}
```

To use these colors, we describe "cycle lists" from which PGF chooses styles for the different series in a chart.

`mbarplot cycle` Colors and styles intended for bar charts with up to 12 series.

```
\pgfplotscreateplotcyclelist{mbarplot cycle}{%
  {draw=TolDarkBlue,    fill=TolDarkBlue!70},
  {draw=TolLightBrown,  fill=TolLightBrown!70},
  {draw=TolLightGreen,  fill=TolLightGreen!70},
  {draw=TolDarkPink,    fill=TolDarkPink!70},
```

[1] Tol actually describes several palettes; these colours are taken from the bottom row of Figure 3 in his technical note.

```
  {draw=TolDarkPurple,  fill=TolDarkPurple!70},
  {draw=TolDarkRed,     fill=TolDarkRed!70},
  {draw=TolDarkBrown,   fill=TolDarkBrown!70},
  {draw=TolLightRed,    fill=TolLightRed!70},
  {draw=TolLightPink,   fill=TolLightPink!70},
  {draw=TolLightPurple, fill=TolLightPurple!70},
  {draw=TolLightBlue,   fill=TolLightBlue!70},
  {draw=TolDarkGreen,   fill=TolDarkGreen!70},
}
```

`mlineplot cycle` Colors and styles intended for line charts with up to 4 series.

```
\pgfplotscreateplotcyclelist{mlineplot cycle}{%
  {TolDarkBlue, mark=*, mark size=1.5pt},
  {TolLightBrown, mark=square*, mark size=1.3pt},
  {TolLightGreen, mark=triangle*, mark size=1.5pt},
  {TolDarkBrown, mark=diamond*, mark size=1.5pt},
}
```

However, the above cycle lists are not applied automatically. We still need to define styles — `mlineplot` and `mbarplot` — that the user can apply to the axis of a `pgfplots` chart to use the colors. We'll also take the opportunity to adjust the display of chart axes when these styles are used.

```
\pgfplotsset{
  compat=1.9,
```

`mlineplot` A style to apply to the axis of a PGF line plot.

```
  mlineplot/.style={
    mbaseplot,
    xmajorgrids=true,
    ymajorgrids=true,
    major grid style={dotted},
    axis x line=bottom,
    axis y line=left,
    legend style={
      cells={anchor=west},
      draw=none
    },
    cycle list name=mlineplot cycle,
},
```

mbarplot
horizontal mbarplot

A style to apply to the axis of a PGF bar chart. mbarplot uses vertical bars by default, while horizontal mbarplot has horizontal bars as the name implies. Their shared properties are factored out into the internal style mbarplot base.

```
mbarplot base/.style={
  mbaseplot,
  bar width=6pt,
  axis y line*=none,
},
mbarplot/.style={
  mbarplot base,
  ybar,
  xmajorgrids=false,
  ymajorgrids=true,
  area legend,
  legend image code/.code={%
    \draw[#1] (0cm,-0.1cm) rectangle (0.15cm,0.1cm);
  },
  cycle list name=mbarplot cycle,
},
horizontal mbarplot/.style={
  mbarplot base,
  xmajorgrids=true,
  ymajorgrids=false,
  xbar stacked,
  area legend,
  legend image code/.code={%
    \draw[#1] (0cm,-0.1cm) rectangle (0.15cm,0.1cm);
  },
  cycle list name=mbarplot cycle,
},
```

mbaseplot

Adjusts the appearance of the axes in a PGF chart.

```
mbaseplot/.style={
  legend style={
    draw=none,
    fill=none,
    cells={anchor=west},
    },
    x tick label style={
      font=\footnotesize
    },
    y tick label style={
      font=\footnotesize
    },
    legend style={
      font=\footnotesize
    },
    major grid style={
      dotted,
    },
    axis x line*=bottom,
  },
  disable thousands separator/.style={
    /pgf/number format/.cd,
      1000 sep={}
  },
}
```