

Modern Beamer Presentations with the **metropolis** package

Matthias Vogelgesang

matthias.vogelgesang@gmail.com

v1.2 — 2017/01/23

Contents

1	Introduction	3
2	Getting Started	4
2.1	Installing from CTAN	4
2.2	Installing from GitHub	4
2.3	A Minimal Example	5
2.4	Dependencies	6
2.5	Pandoc	6
3	Customization	6
3.1	Package options	6
3.1.1	Main theme	7
3.1.2	Inner theme	7
3.1.3	Outer theme	8
3.1.4	Color theme	8
3.1.5	Font theme	8
3.2	Color Customization	8
3.3	Font Customization	9
3.3.1	Old style figures	10
3.4	Commands	10
3.4.1	Standout frames	10

4	pgfplots integration	10
4.1	Styles	10
4.2	Paul Tol colors	11
5	Tips & Tricks	11
5.1	Backup Slides	11
6	Known Issues	12
6.1	Title formats	12
6.2	Interactions with other color themes	12
6.3	Notes on second screen	13
6.4	Standout frames with labels	14
6.5	Standout frames with Pandoc	14
7	License	14
8	Implementation	15
8.1	metropolis parent theme	15
8.1.1	Package dependencies	15
8.1.2	Options	15
8.1.3	Component sub-packages	17
8.1.4	Custom commands	18
8.1.5	Process package options	18
8.2	metropolis inner theme	18
8.2.1	Package dependencies	19
8.2.2	Options	19
8.2.3	Title page	20
8.2.4	Section page	22
8.2.5	Block environments	25
8.2.6	Lists and floats	27
8.2.7	Footnotes	27
8.2.8	Text and spacing settings	27
8.2.9	Standout frames	28
8.2.10	Process package options	29
8.3	metropolis outer theme	29
8.3.1	Package dependencies	30
8.3.2	Options	30
8.3.3	Head and footline	31

8.3.4	Frametitle	32
8.3.5	Process package options	33
8.4	metropolis font theme	33
8.4.1	Package dependencies	34
8.4.2	Load Fira fonts	34
8.4.3	General font definitions	36
8.4.4	Title format options	37
8.4.5	Process package options	43
8.5	metropolis color theme	43
8.5.1	Package dependencies	43
8.5.2	Options	43
8.5.3	Base colors	44
8.5.4	Base styles	44
8.5.5	Derived colors	44
8.5.6	Process package options	47
8.6	Tol pgfplots theme	47

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 can be a little cluttered, while 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 X_YLaTeX to typeset your slides. However, **metropolis** can also be used with other typefaces and L^AT_EX 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 CTAN

For most users, we recommend installing **metropolis** from [CTAN](#). If you keep your \TeX distribution up-to-date, chances are good that **metropolis** is already installed. If it is not, you need to update your packages. If your distribution is \TeX Live (or Mac \TeX on OS X), the following command updates all packages.

```
tlmgr update --all
```

If this results in an error, you may need to run it with administrative privileges:

```
sudo tlmgr update --all
```

Mac \TeX on OS X also provides a graphical interface for **tlmgr** called \TeX Live Utility.

For any other distribution please refer to its documentation on how to update your packages.

To get the most out of the theme you should also install the **Fira** fonts. However, this is not mandatory; **metropolis** also works with the standard fonts.

2.2 Installing from GitHub

If you want to use the cutting-edge development version of **metropolis**, you can install it manually. Like any \LaTeX package, this 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{metropolis}` 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 and manual.

`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.

2.3 A Minimal Example

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

```
\documentclass{beamer}
\usetheme{metropolis}      % 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}
```

2.4 Dependencies

metropolis depends on the **beamer** class and the following standard packages:

- `tikz`
- `etoolbox`
- `ifxetex`
- `pgfopts`
- `calc`
- `ifluatex`

For best results, we recommend installing the fonts **Fira Sans** and **Fira Mono** and compiling with **metropolis** using $\text{Xe}\text{L}\text{A}\text{T}\text{E}\text{X}$ or $\text{Lua}\text{T}\text{E}\text{X}$. These are optional dependencies; **metropolis** is compatible with (e.g.) $\text{pdf}\text{L}\text{A}\text{T}\text{E}\text{X}$ and will fall back to standard fonts if **Fira Sans** or **Fira Mono** is not installed.

The packaged name of **Fira Sans** is **Fira Sans OT** in some Linux distributions; this case is automatically handled by **metropolis**.

2.5 Pandoc

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

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

3 Customization

3.1 Package options

The theme provides a number of options, which can be set using a key=value interface. The primary way to set options is to provide a comma-separated list of option-value pairs when loading **metropolis** in the preamble:

```
\usetheme[option1=value1, option2=value2, ...]{metropolis}
```

Options can be changed at any time — even mid-presentation! — with the `\metroset` macro.

```
\metroset{option1=newvalue1, option2=newvalue2, ...}
```

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

option key *list of possible values* default
 A short description of the option.

3.1.1 Main theme

titleformat *regular, smallcaps, allsmallcaps, allcaps* regular
 Changes the format of titles, subtitles, section titles, frame titles, and the text on “standout” frames. The available options produce Regular, SMALLCAPS, ALLSMALLCAPS, or ALLCAPS titles. Please refer to Section 6.1 for known issues with these options.

titleformat plain *regular, smallcaps, allsmallcaps, allcaps* regular
 Changes the format of “standout” frames (see titleformat, above).

3.1.2 Inner theme

sectionpage *none, simple, progressbar* progressbar
 Adds a slide at the start of each section (**simple**) with an optional thin progress bar below the section title (**progressbar**). The **none** option disables the section page.

subsectionpage *none, simple, progressbar* none
 Optionally adds a slide at the start of each subsection. If enabled with the **simple** or **progressbar** options, the style of the **section page** will be updated to match the style of the **subsection page**. Note that section slides and subsection slides can appear consecutively if both are enabled; you may want to use this option together with **sectionpage=none** depending on the section structure of your presentation.

3.1.3 Outer theme

numbering	<i>none, counter, fraction</i>	counter
	Controls whether the frame number at the bottom right of each slide is omitted (none), shown (counter) or displayed as a fraction of the total number of frames (fraction).	
progressbar	<i>none, head, frametitle, foot</i>	none
	Optionally adds a progress bar to the top of each frame (head), the bottom of each frame (foot), or directly below each frame title (frametitle).	

3.1.4 Color theme

block	<i>transparent, fill</i>	transparent
	Optionally adds a light grey background to block environments like theorem and example .	
background	<i>dark, light</i>	light
	Provides the option to have a dark background and light foreground instead of the reverse.	

3.1.5 Font theme

titleformat title	<i>regular, smallcaps, allsmallcaps, allcaps</i>	regular
titleformat subtitle	Individually controls the format of titles, subtitles, section titles, and frame titles (see titleformat , above).	
titleformat section		
titleformat frame		

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}{ ... }
```

For low-light situations **metropolis** it might be helpful to use the **metropolis-highcontrast** color theme. It is enabled like any other color theme:

```
\usecolortheme{metropolis-highcontrast}
```

3.3 Font Customization

The default font for **metropolis** is Fira. This can be easily changed using the standard font selection commands of the **fontspec** package. So if you prefer, for example, the **Ubuntu** font family, just add the following two commands after loading the **metropolis** theme.

```
\setsansfont{Ubuntu}
\setmonofont{Ubuntu Mono}
```

If you are expecting to present in a large room or with an underpowered projector, you may want to change the font to a heavier weight of Fira to maximize readability.

```
\setsansfont[BoldFont={Fira Sans SemiBold}]{Fira Sans Book}
```

3.3.1 Old style figures

The regular `fontspec` mechanism for changing glyph appearance applies also to this theme. If you want to have old style figures in the text but regular lined figures for math, you could add the following to your preamble:

```
\usefonttheme{professionalfonts}    % required for mathspec
\usepackage{mathspec}
\setsansfont[BoldFont={Fira Sans},
             Numbers={OldStyle}]{Fira Sans Light}
\setmathsfon(Digits)[Numbers={Lining, Proportional}]{Fira
             Sans Light}
```

3.4 Commands

3.4.1 Standout frames

The **metropolis** inner theme offers a custom frame format with large, centered text and an inverted background — perfect for focusing attention on single sentence or image. To use it, add the key `standout` to the frame:

```
\begin{frame}[standout]
    Thank you!
\end{frame}
```

4 pgfplots integration

metropolis comes with a set of pre-defined `pgfplots` styles and a color theme based on Paul Tol's color scheme.

4.1 Styles

Pass the following style keys to the axis environment to get the appropriate effect:

<code>mlineplot</code>	Plot regular line charts with reduced axis frames, less intrusive legend and subdued grid.
<code>mbarplot</code>	Plot vertical bar charts in a similar way as <code>mlineplot</code> but reduce grid usage.
<code>horizontal mbarplot</code>	Plot horizontal bar charts.
<code>disable thousands separator</code>	Helper style to remove thousands separator.

4.2 Paul Tol colors

A good presentation uses colors that are distinct from each other as much as possible as well as from black and white, can be discerned item under different lighting and display environments and by color-blind viewers, while matching well together.

In a [technical note](#) for SRON, Paul Tol proposed a palette of colors satisfying these constraints. The sub-package `pgfplotsmetol` defines palettes for `pgfplots` charts based on Tol's work.

5 Tips & Tricks

5.1 Backup Slides

Speakers will often include extra slides at the end of their presentation to refer to during audience questions. One easy way to do this is to include the `appendixnumberbeamer` package in your preamble and call `\appendix` before your backup slides.

metropolis will automatically turn off slide numbering and progress bars for slides in the appendix.

6 Known Issues

6.1 Title formats

Be aware that not every font supports small caps, so the `smallcaps` or `allsmallcaps` options may not work if you use a font other than Fira Sans. In particular, the Computer Modern sans-serif typeface, which is used when **metropolis** is compiled with pdfL^AT_EX, does not have a small-caps variant.

The title format options `allsmallcaps` and `allcaps` are quite nice from an aesthetic point of view, but their use of `\MakeLowercase` and `\MakeUppercase` can cause unexpected problems. For example:

- Some commands, like `\`, do not work inside `\MakeLowercase` and `\MakeUppercase`. (See [#125](#))
- Only alphabetic characters are affected by `\MakeLowercase`, so numerals and punctuation remain at full height. This can spoil some of the aesthetic benefits of `allsmallcaps`. (See [#33](#))
- `\MakeLowercase` and `\MakeUppercase` apply to math mode and `\scshape` does not. This can easily introduce mathematical errors that are hard to catch.
- It is impossible to typeset symbols which are encoded as uppercase letters in a different font. In particular, `\mathbb` and `\mathcal` letters will be replaced by other math glyphs. (See [#153](#))

The `allsmallcaps` and `allcaps` options are safe to use if your titles contain only alphabetic characters and do not require the expansion of any macros.

6.2 Interactions with other color themes

metropolis can be used along with any other Beamer color theme, such as **crane** or **seahorse**. If you wish to do this, it is usually best to include the **metropolis** subpackages individually so the **metropolis** color theme is never loaded. This will prevent conflicts between the **metropolis** color theme and your preferred theme.

For example, overriding the color theme as follows may not work as expected because `\usetheme{metropolis}` loads the **metropolis** color theme, which defines a relationship between the frametitle background and the primary palette of the

theme. Since **seahorse** assumes a different relationship between its palettes, the result is a grey, rather than periwinkle, frametitle background.

```
\usetheme{metropolis}  
\usecolortheme{seahorse}
```

The correct colors are chosen if the **metropolis** outer, inner, and font themes are loaded separately:

```
\useoutertheme{metropolis}  
\useinnertheme{metropolis}  
\usefonttheme{metropolis}  
\usecolortheme{seahorse}    % or your preferred color theme
```

Please note that **metropolis** may not use all the colors defined in your favourite Beamer color theme. In particular, **metropolis** does not set a background color for the title; this will cause issues when using color themes like **whale** which set a white foreground for the title.

6.3 Notes on second screen

If you use the `[show notes on second screen]` option built in to Beamer and compile with $\text{X}\text{\LaTeX}$, text on slides following the first section slide may be rendered in white instead of the regular colour. This is due to a bug in Beamer or $\text{X}\text{\LaTeX}$ itself. You can work around it either by compiling with $\text{Lua}\text{\TeX}$ or by adding the following code to your preamble to reset the text color on each slide.

```
\makeatletter  
\def\beamer@framenotesbegin{% at beginning of slide  
  \usebeamercolor[fg]{normal text}  
  \gdef\beamer@noteitems{}%  
  \gdef\beamer@notes{}%  
}  
\makeatother
```

6.4 Standout frames with labels

Because the `standout` frame option creates a group to restrict the colour change to a single slide, labels defined after calling `standout` will stay local to the group. In other words, the following may result in a “label undefined” error.

```
\begin{frame}[standout, label=conclusion]{Conclusion}
  Awesome slide
\end{frame}
```

To fix this problem, change the order of the keys in the frame.

```
\begin{frame}[label=conclusion, standout]{Conclusion}
  Awesome slide
\end{frame}
```

This error can be unwittingly triggered if you export your slides from Emacs Org mode, which automatically adds labels after frame options. Alex Branham [offers](#) the following solution for Org mode users, using `org-set-property`.

```
* Start of a frame
:PROPERTIES:
:BEAMER_opt: label=conclusion,standout
:END:
```

6.5 Standout frames with Pandoc

With Pandoc versions prior 1.17.2 it was not possible to create standout frames because Pandoc only supported a specific list of frame attributes thus ignoring additional attributes such as `{.standout}`.

7 License

metropolis 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 any presentations that you create with the theme.

8 Implementation

8.1 metropolis parent 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.

8.1.1 Package dependencies

```
1 \RequirePackage{etoolbox}
2 \RequirePackage{pgfopts}
```

8.1.2 Options

Most options are passed off to the component sub-packages.

```
3 \pgfkeys{/metropolis/.cd,
4   .search also={
5     /metropolis/inner,
6     /metropolis/outer,
7     /metropolis/color,
8     /metropolis/font,
9   }
10 }
```

`titleformat plain` Controls the formatting of the text on standout “plain” frames.

```
11 \pgfkeys{
12   /metropolis/titleformat plain/.cd,
13   .is choice,
14   regular/.code={%
15     \let\metropolis@plaintitleformat\@empty%
16     \setbeamerfont{standout}{shape=\normalfont}%
17   },
```

```

18     smallcaps/.code={%
19         \let\metropolis@plaintitleformat\@empty%
20         \setbeamerfont{standout}{shape=\scshape}%
21     },
22     allsmallcaps/.code={%
23         \let\metropolis@plaintitleformat\MakeLowercase%
24         \setbeamerfont{standout}{shape=\scshape}%
25         \PackageWarning{beamerthememetropolis}{%
26             Be aware that titleformat plain=allsmallcaps can lead to problems%
27         }
28     },
29     allcaps/.code={%
30         \let\metropolis@plaintitleformat\MakeUppercase%
31         \setbeamerfont{standout}{shape=\normalfont}%
32         \PackageWarning{beamerthememetropolis}{%
33             Be aware that titleformat plain=allcaps can lead to problems%
34         }
35     },
36 }

```

titleformat Sets a standard format for titles, subtitles, section titles, frame titles, and the text on standout “plain” frames.

```

37 \pgfkeys{
38   /metropolis/titleformat/.code=\pgfkeysalso{
39     font/titleformat title=#1,
40     font/titleformat subtitle=#1,
41     font/titleformat section=#1,
42     font/titleformat frame=#1,
43     titleformat plain=#1,
44   }
45 }

```

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

```

46 \pgfkeys{/metropolis/.cd,
47   usetitleprogressbar/.code=\pgfkeysalso{outer/progressbar=frametitle},
48   noslidenumbers/.code=\pgfkeysalso{outer/numbering=none},
49   usetotalslideindicator/.code=\pgfkeysalso{outer/numbering=fraction},
50   nosectionslide/.code=\pgfkeysalso{inner/sectionpage=none},

```



```

51 darkcolors/.code=\pgfkeysalso{color/background=dark},
52 blockbg/.code=\pgfkeysalso{color/block=fill, inner/block=fill},
53 }

```

Set default values for options.

```

54 \newcommand{\metropolis@setdefaults}{
55   \pgfkeys{/metropolis/.cd,
56     titleformat plain=regular,
57   }
58 }

```

To avoid generating externalized figures of the progressbar we have to disable them with “tikzexternalenable” and “tikzexternaldisable”. However, if the “external” library is not loaded we would get undefined control sequence problems, hence we define them as no-ops if they are not defined yet.

```

59 \providecommand{\tikzexternalenable}{}
60 \providecommand{\tikzexternaldisable}{}

```

8.1.3 Component sub-packages

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

```

61 \useinnertheme{metropolis}
62 \useoutertheme{metropolis}
63 \usecolortheme{metropolis}
64 \usefonttheme{metropolis}

```

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

```

65 \AtEndPreamble{%
66   \ifpackageloaded{pgfplots}{%
67     \RequirePackage{pgfplotsthemetol}
68   }{}
69 }

```

8.1.4 Custom commands

The parent theme defines custom commands as their proper usage may depend on multiple sub-packages.

`\metroset` Allows the user to change options midway through a presentation.

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

`\plain` Creates a plain frame with dark background, suitable for displaying images or a few words. The format of the text can be set with the `titleformat plain` option.

```
71 \def\metropolis@plaintitleformat#1{#1}
72 \newcommand{\plain}[2][]{%
73   \PackageWarning{beamerthememetropolis}{%
74     The syntax '\plain' may be deprecated in a future version of Metropolis.
75     Please use a frame with [standout] instead.
76   }
77   \begin{frame}[standout]{#1}
78     \metropolis@plaintitleformat{#2}
79   \end{frame}
80 }
```

`\mreducelistspacing`

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

8.1.5 Process package options

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

8.2 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.

8.2.1 Package dependencies

```
84 \RequirePackage{etoolbox}
85 \RequirePackage{keyval}
86 \RequirePackage{calc}
87 \RequirePackage{pgfopts}
88 \RequirePackage{tikz}
```

8.2.2 Options

sectionpage Optionally add a slide marking the beginning of each section.

```
89 \pgfkeys{
90   /metropolis/inner/sectionpage/.cd,
91   .is choice,
92   none/.code=\metropolis@disablesectionpage,
93   simple/.code={\metropolis@enablesectionpage
94                 \setbeamertemplate{section page}[simple]},
95   progressbar/.code={\metropolis@enablesectionpage
96                      \setbeamertemplate{section page}[progressbar]},
97 }
```

subsectionpage Optionally add a slide marking the beginning of each subsection.

```
98 \pgfkeys{
99   /metropolis/inner/subsectionpage/.cd,
100   .is choice,
101   none/.code=\metropolis@disablesubsectionpage,
102   simple/.code={\metropolis@enablesubsectionpage
103                 \setbeamertemplate{section page}[simple]},
104   progressbar/.code={\metropolis@enablesubsectionpage
105                      \setbeamertemplate{section page}[progressbar]},
106 }
```

\metropolis@inner@setdefaults Set default values for inner theme options.

```
107 \newcommand{\metropolis@inner@setdefaults}{
108   \pgfkeys{/metropolis/inner/.cd,
109     sectionpage=progressbar,
```

```

110     subsectionpage=none
111 }
112 }

```

8.2.3 Title page

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.

```

113 \setbeamertemplate{title page}{
114   \begin{minipage}[b][\paperheight]{\textwidth}
115     \ifx\inserttitlegraphic\@empty\else\usebeamertemplate*{title graphic}\fi
116     \vfill%
117     \ifx\inserttitle\@empty\else\usebeamertemplate*{title}\fi
118     \ifx\insertsubtitle\@empty\else\usebeamertemplate*{subtitle}\fi
119     \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](#).

```

120     \ifx\beamer@shortauthor\@empty\else\usebeamertemplate*{author}\fi
121     \ifx\insertdate\@empty\else\usebeamertemplate*{date}\fi
122     \ifx\insertinstitute\@empty\else\usebeamertemplate*{institute}\fi
123     \vfill
124     \vspace*{1mm}
125   \end{minipage}
126 }

```

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

```

\titlepage
127 \def\maketitle{%
128   \ifbeamer@inframe

```

```

129   \titlepage
130   \else
131   \frame[plain,noframenumbering]{\titlepage}
132   \fi
133 }
134 \def\titlepage{%
135   \usebeamertemplate{title page}
136 }

```

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

```

137 \setbeamertemplate{title graphic}{
138   \vbox to 0pt {
139     \vspace*{2em}
140     \inserttitlegraphic%
141   }%
142   \nointerlineskip%
143 }

```

title Set the title on the title page.

```

144 \setbeamertemplate{title}{
145   \raggedright%
146   \linespread{1.0}%
147   \inserttitle%
148   \par%
149   \vspace*{0.5em}
150 }

```

subtitle Set the subtitle on the title page.

```

151 \setbeamertemplate{subtitle}{
152   \raggedright%
153   \insertsubtitle%
154   \par%
155   \vspace*{0.5em}
156 }

```

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

```

157 \newlength{\metropolis@titleseparator@linewidth}
158 \setlength{\metropolis@titleseparator@linewidth}{0.4pt}
159 \setbeamertemplate{title separator}{
160   \tikzexternaldisable%
161   \begin{tikzpicture}
162     \fill[fg] (0,0) rectangle (\textwidth, \metropolis@titleseparator@linewidth);
163   \end{tikzpicture}%
164   \tikzexternalenable%
165   \par%
166 }

```

author Set the author on the title page.

```

167 \setbeamertemplate{author}{
168   \vspace*{2em}
169   \insertauthor%
170   \par%
171   \vspace*{0.25em}
172 }

```

date Set the date on the title page.

```

173 \setbeamertemplate{date}{
174   \insertdate%
175   \par%
176 }

```

institute Set the institute on the title page.

```

177 \setbeamertemplate{institute}{
178   \vspace*{3mm}
179   \insertinstitute%
180   \par%
181 }

```

8.2.4 Section page

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

```

182 \defbeamertemplate{section page}{simple}{
183   \begin{center}

```

```

184 \usebeamercolor[fg]{section title}
185 \usebeamerfont{section title}
186 \insertsectionhead\par
187 \ifx\insertsubsectionhead\@empty\else
188 \usebeamercolor[fg]{subsection title}
189 \usebeamerfont{subsection title}
190 \insertsubsectionhead
191 \fi
192 \end{center}
193 }
194 \defbeamertemplate{section page}{progressbar}{
195 \centering
196 \begin{minipage}{22em}
197 \raggedright
198 \usebeamercolor[fg]{section title}
199 \usebeamerfont{section title}
200 \insertsectionhead\[-1ex]
201 \usebeamertemplate*{progress bar in section page}
202 \par
203 \ifx\insertsubsectionhead\@empty\else%
204 \usebeamercolor[fg]{subsection title}%
205 \usebeamerfont{subsection title}%
206 \insertsubsectionhead
207 \fi
208 \end{minipage}
209 \par
210 \vspace{\baselineskip}
211 }
212 \newcommand{\metropolis@disablesectionpage}{
213 \AtBeginSection{
214 % intentionally empty
215 }
216 }
217 \newcommand{\metropolis@enablesectionpage}{
218 \AtBeginSection{
219 \ifbeamer@inframe
220 \sectionpage
221 \else
222 \frame[plain,c,noframenumbers]{\sectionpage}
223 \fi

```

```

224 }
225 }

```

subsection page Template for the subsection title slide that can optionally be added to at the beginning of each subsection.

```

226 \setbeamertemplate{subsection page}{%
227   \usebeamertemplate*{section page}
228 }
229 \newcommand{\metropolis@disablesubsectionpage}{
230   \AtBeginSubsection{
231     % intentionally empty
232   }
233 }
234 \newcommand{\metropolis@enablesubsectionpage}{
235   \AtBeginSubsection{
236     \ifbeamer@inframe
237       \subsectionpage
238     \else
239       \frame[plain,c,noframenumbering]{\subsectionpage}
240     \fi
241   }
242 }

```

progress 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`.

```

243 \newlength{\metropolis@progressonsectionpage}
244 \newlength{\metropolis@progressonsectionpage@linewidth}
245 \setlength{\metropolis@progressonsectionpage@linewidth}{0.4pt}
246 \setbeamertemplate{progress bar in section page}{
247   \setlength{\metropolis@progressonsectionpage}{%
248     \textwidth * \ratio{\insertframenum pt}{\inserttotalframenum pt}}%
249   }%
250   \tikzexternaldisable%
251   \begin{tikzpicture}
252     \fill[bg] (0,0) rectangle (\textwidth, \metropolis@progressonsectionpage@linewidth);
253     \fill[fg] (0,0) rectangle (\metropolis@progressonsectionpage, \metropolis@progressonsectionpage@linewidth);
254   \end{tikzpicture}%
255   \tikzexternalenable%
256 }

```


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@progressonsectionpage` 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.

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

8.2.5 Block environments

`block` The three different block environments differ only in their colours. Rather than
`block alerted` repeat the essentially the same template three times, we use the auxiliary macro
`block example` `\metropolis@block` to define all three templates.

```
258 \newlength{\metropolis@blocksep}
259 \newlength{\metropolis@blockadjust}
260 \setlength{\metropolis@blocksep}{0.75ex}
261 \setlength{\metropolis@blockadjust}{0.25ex}
262 \providecommand{\metropolis@strut}{%
263   \vphantom{ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz}}%
264 }
265 \newcommand{\metropolis@block}[1]{
266   \par\vskip\medskipamount%
267   \setlength{\parskip}{0pt}
```

If a background color is defined for the block title or body, we need to add a little bit of padding to the corresponding box. Ideally, this would be accomplished by setting `colsep=0.75ex`, which is intended to add “color separation space” only when the box has a colored background. Unfortunately, `colsep` also adds this separation if the background color is inherited, even if the inherited color is actually empty. (The technical reason for this boils down to the fact that the `\ifx` directive does not expand macros.)

To achieve the correct spacing for `alertblocks` and `exampleblocks` as well as for normal blocks, we have to begin the `beamercolorbox` differently based on whether `block title` has an empty background.

If the `block title` background is empty, or the user has explicitly removed the background from (e.g.) `block title alerted`, we just need to set a `rightskip` for a nice ragged-right block title.

```

268 \ifbeamercoloreempty[bg]{block title#1}{%
269   \begin{beamercolorbox}[rightskip=0pt plus 4em]{block title#1}}{%
270 \ifbeamercoloreempty[bg]{block title}{%
271   \begin{beamercolorbox}[rightskip=0pt plus 4em]{block title#1}%
272   }%
273 %   \end{macrocode}
274 %
275 %   Otherwise, if the |block title| has a background, we set the padding based
276 %   on |\metropolis@blockskip|. However, we have to visually compensate for
277 %   the |\metropolis@strut| added to the block title (see below) by
278 %   subtracting |\metropolis@blockadjust| from the top and bottom padding.
279 %
280 %   \begin{macrocode}
281 {%
282   \begin{beamercolorbox}[
283     sep=\dimexpr\metropolis@blocksep-\metropolis@blockadjust\relax,
284     leftskip=\metropolis@blockadjust,
285     rightskip=\dimexpr\metropolis@blockadjust plus 4em\relax
286   ]{block title#1}%
287   }%
288 %   \end{macrocode}
289 %
290 %   We can now set the contents of the |block title|. The zero-width but
291 %   positive-height box |\metropolis@strut| ensures that the block title box
292 %   has a consistent height, even if it lacks punctuation, ascenders, or
293 %   descenders.
294 %
295 %   \begin{macrocode}
296     \usebeamerfont*{block title#1}%
297     \metropolis@strut%
298     \insertblocktitle%
299     \metropolis@strut%
300 \end{beamercolorbox}%
301 %   \end{macrocode}
302 %
303 %   Next, we typeset the |block body|. This the code is similar to, but simpler

```

```

304 %   than, the |block title| code since we don't need to adjust for any struts.
305 %
306 %   \begin{macrocode}
307 \nointerlineskip%
308 \ifbeamercolorempy[bg]{block body#1}{%
309   \begin{beamercolorbox}[vmode]{block body#1}}{
310 \ifbeamercolorempy[bg]{block body}{%
311   \begin{beamercolorbox}[vmode]{block body#1}%
312 }{%
313   \begin{beamercolorbox}[sep=\metropolis@blocksep, vmode]{block body#1}%
314   \vspace{-\metropolis@parskip}
315 }}%
316   \usebeamerfont{block body#1}%
317   \setlength{\parskip}{\metropolis@parskip}%
318 }

```

This concludes the auxiliary macro `\metropolis@block`. Finally, we define the block beamer templates using this macro.

```

319 \setbeamertemplate{block begin}{\metropolis@block{}}
320 \setbeamertemplate{block alerted begin}{\metropolis@block{ alerted}}
321 \setbeamertemplate{block example begin}{\metropolis@block{ example}}
322 \setbeamertemplate{block end}{\end{beamercolorbox}\vspace*{0.2ex}}
323 \setbeamertemplate{block alerted end}{\end{beamercolorbox}\vspace*{0.2ex}}
324 \setbeamertemplate{block example end}{\end{beamercolorbox}\vspace*{0.2ex}}

```

8.2.6 Lists and floats

```

325 \setbeamertemplate{itemize items}{\textbullet}
326 \setbeamertemplate{caption label separator}{: }
327 \setbeamertemplate{caption}[numbered]

```

8.2.7 Footnotes

```

328 \setbeamertemplate{footnote}{%
329   \parindent 0em\noindent%
330   \raggedright
331   \usebeamercolor{footnote}\hbox to 0.8em{\hfil\insertfootnotemark}\insertfootnotetext\par%
332 }

```

8.2.8 Text and spacing settings

```

333 \newlength{\metropolis@parskip}
334 \setlength{\metropolis@parskip}{0.5em}
335 \setlength{\parskip}{\metropolis@parskip}
336 \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](#).

```

337 \define@key{beamerframe}{c}[true]{% centered
338   \beamer@frametopskip=0pt plus 1fill\relax%
339   \beamer@framebottomskip=0pt plus 1fill\relax%
340   \beamer@frametopskipautobreak=0pt plus .4\paperheight\relax%
341   \beamer@framebottomskipautobreak=0pt plus .6\paperheight\relax%
342   \def\beamer@initfirstlineunskip{}%
343 }

```

8.2.9 Standout frames

metropolis offers a custom frame format with large, centered text and an inverted background. To use it, add the key `standout` to the frame: `\begin{frame}[standout] ... \end{frame}`.

standout Optional arguments to Beamer's frames are implemented using `\define@key` from the `keyval` package, which will execute code when the defined option is called. For the `standout` option, we begin a group, change the colors and set frame options.

```

344 \providebool{metropolis@standout}
345 \define@key{beamerframe}{standout}[true]{%
346   \booltrue{metropolis@standout}
347   \begingroup
348     \setkeys{beamerframe}{c}
349     \setkeys{beamerframe}{noframenumbering}
350     \ifbeamercoloreempty[bg]{palette primary}{
351       \setbeamercolor{background canvas}{
352         use=palette primary,
353         bg=-palette primary.fg
354       }
355     }{
356       \setbeamercolor{background canvas}{
357         use=palette primary,

```

```

358         bg=palette primary.bg
359     }
360 }
361 \setbeamercolor{local structure}{
362     fg=palette primary.fg
363 }
364 \usebeamercolor[fg]{palette primary}
365 }

```

Then we just have to close the group after the standout slide is finished in order to restore the colours and fonts for the rest of the presentation. Unfortunately, we cannot use or this (see <http://tex.stackexchange.com/questions/226319/>). Instead, we prepend the `\endgroup` to `\beamer@reseteecodes`, which is run exactly once at the end of each slide.

```

366 \pretocmd{\beamer@reseteecodes}{%
367     \ifbool{metropolis@standout}{
368         \endgroup
369         \boolfalse{metropolis@standout}
370     }{}
371 }{}{}

```

We set the fonts and the alignment on the inner content, in such a way that the speaker's note layout isn't affected by the custom formatting.

```

372 \AtBeginEnvironment{beamer@frameslide}{
373     \ifbool{metropolis@standout}{
374         \centering
375         \usebeamerfont{standout}
376     }{}
377 }

```

8.2.10 Process package options

```

378 \metropolis@inner@setdefaults
379 \ProcessPgfPackageOptions{/metropolis/inner}

```

8.3 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.

8.3.1 Package dependencies

```
380 \RequirePackage{etoolbox}
381 \RequirePackage{calc}
382 \RequirePackage{pgfopts}
```

8.3.2 Options

numbering Adds slide numbers to the bottom right of each slide.

```
383 \pgfkeys{
384   /metropolis/outer/numbering/.cd,
385   .is choice,
386   none/.code=\setbeamertemplate{frame numbering}[none],
387   counter/.code=\setbeamertemplate{frame numbering}[counter],
388   fraction/.code=\setbeamertemplate{frame numbering}[fraction],
389 }
```

progressbar Adds a progress bar to the top, bottom, or frametitle of each slide.

```
390 \pgfkeys{
391   /metropolis/outer/progressbar/.cd,
392   .is choice,
393   none/.code={%
394     \setbeamertemplate{headline}[plain]
395     \setbeamertemplate{frametitle}[plain]
396     \setbeamertemplate{footline}[plain]
397   },
398   head/.code={\pgfkeys{/metropolis/outer/progressbar=none}
399     \addtobeamertemplate{headline}{}{%
400       \usebeamertemplate*{progress bar in head/foot}
401     }
402   },
403   frametitle/.code={\pgfkeys{/metropolis/outer/progressbar=none}
404     \addtobeamertemplate{frametitle}{}{%
405       \usebeamertemplate*{progress bar in head/foot}
406     }
407   },
408   foot/.code={\pgfkeys{/metropolis/outer/progressbar=none}
409     \addtobeamertemplate{footline}{}{%
410       \usebeamertemplate*{progress bar in head/foot}%
411     }
412   },
413 }
```

```

411     }
412   },
413 }

```

`\metropolis@outer@setdefaults` Sets default values for outer theme options.

```

414 \newcommand{\metropolis@outer@setdefaults}{
415   \pgfkeys{/metropolis/outer/.cd,
416     numbering=counter,
417     progressbar=none,
418   }
419 }

```

8.3.3 Head and footline

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

```

420 \setbeamertemplate{navigation symbols}{}

```

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

```

421 \defbeamertemplate{frame footer}{none}{}
422 \defbeamertemplate{frame footer}{custom}[1]{ #1 }

423 \defbeamertemplate{frame numbering}{none}{}
424 \defbeamertemplate{frame numbering}{counter}{\insertframenumbers}
425 \defbeamertemplate{frame numbering}{fraction}{
426   \insertframenumbers/\inserttotalframenumbers
427 }

```

headline Templates for the head- and footline at the top and bottom of each frame.

footline

```

428 \defbeamertemplate{headline}{plain}{}
429 \defbeamertemplate{footline}{plain}{%
430   \begin{beamercolorbox}[wd=\textwidth, sep=3ex]{footline}%
431     \usebeamerfont{page number in head/foot}%
432     \usebeamertemplate*{frame footer}
433     \hfill%
434     \usebeamertemplate*{frame numbering}

```

```

435 \end{beamercolorbox}%
436 }

```

8.3.4 Frametitle

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

```

437 \newlength{\metropolis@frametitle@padding}
438 \setlength{\metropolis@frametitle@padding}{2.2ex}
439 \newcommand{\metropolis@frametitlestrut@start}{
440   \rule{0pt}{\metropolis@frametitle@padding +%
441     \totalheightof{
442       \ifcsdef{metropolis@frametitleformat}{\metropolis@frametitleformat X}{X}%
443     }%
444   }%
445 }
446 \newcommand{\metropolis@frametitlestrut@end}{
447   \rule[-\metropolis@frametitle@padding]{0pt}{\metropolis@frametitle@padding}
448 }
449 \defbeamertemplate{frametitle}{plain}{%
450   \nointerlineskip%
451   \begin{beamercolorbox}[%
452     wd=\paperwidth,%
453     sep=0pt,%
454     leftskip=\metropolis@frametitle@padding,%
455     rightskip=\metropolis@frametitle@padding,%
456   ]{frametitle}%
457   \metropolis@frametitlestrut@start%
458   \insertframetitle%
459   \nolinebreak%
460   \metropolis@frametitlestrut@end%
461   \end{beamercolorbox}%
462 }
463 \setbeamertemplate{frametitle continuation}{%
464   \usebeamerfont{frametitle}
465   \romannumeral \insertcontinuationcount
466 }

```

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.

```

467 \newlength{\metropolis@progressinheadfoot}
468 \newlength{\metropolis@progressinheadfoot@linewidth}
469 \setlength{\metropolis@progressinheadfoot@linewidth}{0.4pt}
470 \setbeamertemplate{progress bar in head/foot}{
471   \nointerlineskip
472   \setlength{\metropolis@progressinheadfoot}{%
473     \paperwidth * \ratio{\insertframenum pt}{\inserttotalframenum pt}}%
474   }%
475   \begin{beamercolorbox}[wd=\paperwidth]{progress bar in head/foot}
476     \tikzexternaldisable%
477     \begin{tikzpicture}
478       \fill[bg] (0,0) rectangle (\paperwidth, \metropolis@progressinheadfoot@linewidth);
479       \fill[fg] (0,0) rectangle (\metropolis@progressinheadfoot, \metropolis@progressinheadfoot);
480     \end{tikzpicture}%
481     \tikzexternalenable%
482   \end{beamercolorbox}
483 }
```

appendix Removes page numbering and per-slide progress bars when `\appendix` is called. This makes it easier to include additional “backup slides” at the end of the presentation, especially in conjunction with the package `appendixnumberbeamer`.

```

484 \AtBeginDocument{%
485   \apptocmd{\appendix}{%
486     \pgfkeys{%
487       /metropolis/outer/.cd,
488       numbering=none,
489       progressbar=none}
490   }{}{}
491 }
```

8.3.5 Process package options

```

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

8.4 metropolis font theme

A beamer font theme sets the style of the font used in the document.

8.4.1 Package dependencies

```
494 \RequirePackage{etoolbox}
495 \RequirePackage{ifxetex}
496 \RequirePackage{ifluatex}
497 \RequirePackage{pgfopts}
```

8.4.2 Load Fira fonts

If the presentation is compiled with Xe \LaTeX or Lua \LaTeX , the fontspec package is loaded and we search for the Fira fonts.

```
498 \ifboolexpr{bool {xetex} or bool {luatex}}{
499   \@ifpackageloaded{fontspec}{
500     \PassOptionsToPackage{no-math}{fontspec}
501   }{
502     \RequirePackage[no-math]{fontspec}
503   }
```

`\checkfont` Checks if a font is installed; if not, `fontsnofound` is increased.

```
504 \newcounter{fontsnofound}
505 \newcommand{\checkfont}[1]{%
506   \suppressfontnotfounderror=1%
507   \font\x = "#1" at 10pt
508   \selectfont
509   \ifx\x\nullfont%
510     \stepcounter{fontsnofound}%
511   \fi%
512   \suppressfontnotfounderror=0%
513 }
514
```

`\iffontsavailable` Resets the `fontsnofound` counter and calls `\checkfont` for each font in the comma separated list in the first argument.

```
515 \newcommand{\iffontsavailable}[3]{%
516   \setcounter{fontsnofound}{0}%
517   \expandafter\forcsvlist\expandafter%
518   \checkfont\expandafter{#1}%
519   \ifnum\value{fontsnofound}=0%
520     #2%
```

```

521 \else%
522     #3%
523 \fi%
524 }

```

We search for regular, italic, light, light italic, mono, and mono bold fonts under the default `Fira Sans` and `Fira Mono` names. If this fails, the suffix `OT` — used by some Linux distributions — will be tried. If this also fails, a warning will be displayed and the standard fonts will be used.

```

525 \iffontsavailable{Fira Sans Light,%
526             Fira Sans Light Italic,%
527             Fira Sans,%
528             Fira Sans Italic}%
529 {%
530     \setsansfont[ItalicFont={Fira Sans Light Italic},%
531             BoldFont={Fira Sans},%
532             BoldItalicFont={Fira Sans Italic}]]%
533             {Fira Sans Light}%
534 }{%
535     \iffontsavailable{Fira Sans Light OT,%
536             Fira Sans Light Italic OT,%
537             Fira Sans OT,%
538             Fira Sans Italic OT}%
539     {%
540         \setsansfont[ItalicFont={Fira Sans Light Italic OT},%
541             BoldFont={Fira Sans OT},%
542             BoldItalicFont={Fira Sans Italic OT}]]%
543             {Fira Sans Light OT}%
544     }{%
545         \PackageWarning{beamerthememetropolis}{%
546             Could not find Fira Sans fonts%
547         }
548     }
549 }
550 \iffontsavailable{Fira Mono, Fira Mono Bold}{%
551     \setmonofont[BoldFont={Fira Mono Medium}]{Fira Mono}%
552 }{%
553     \iffontsavailable{Fira Mono OT, Fira Mono Bold OT}{%
554         \setmonofont[BoldFont={Fira Mono Medium OT}]{Fira Mono OT}%

```

```

555     }{%
556         \PackageWarning{beamerthememetropolis}{%
557             Could not find Fira Mono fonts%
558         }
559     }
560 }
561 \AtBeginEnvironment{tabular}{%
562     \addfontfeature{Numbers={Monospaced}}}%
563 }
564 }{%
565     \PackageWarning{beamerthememetropolis}{%
566         You need to compile with XeLaTeX or LuaLaTeX to use the Fira fonts%
567     }
568 }

```

This concludes the portion of the code which is only run when compiled with Xe_ΛTeX or Lua_ΛTeX. The remainder of this package applies regardless of the compiling engine.

8.4.3 General font definitions

```

569 \setbeamerfont{title}{size=\Large,%
570                 series=\bfseries}
571 \setbeamerfont{author}{size=\small}
572 \setbeamerfont{date}{size=\small}
573 \setbeamerfont{section title}{size=\Large,%
574                 series=\bfseries}
575 \setbeamerfont{block title}{size=\normalsize,%
576                 series=\bfseries}
577 \setbeamerfont{block title alerted}{size=\normalsize,%
578                 series=\bfseries}
579 \setbeamerfont*{subtitle}{size=\large}
580 \setbeamerfont{frametitle}{size=\large,%
581                 series=\bfseries}
582 \setbeamerfont{caption}{size=\small}
583 \setbeamerfont{caption name}{series=\bfseries}
584 \setbeamerfont{description item}{series=\bfseries}
585 \setbeamerfont{page number in head/foot}{size=\scriptsize}
586 \setbeamerfont{bibliography entry author}{size=\normalsize,%
587                 series=\normalfont}

```

```

588 \setbeamerfont{bibliography entry title}{size=\normalsize,%
589                                     series=\bfseries}
590 \setbeamerfont{bibliography entry location}{size=\normalsize,%
591                                     series=\normalfont}
592 \setbeamerfont{bibliography entry note}{size=\small,%
593                                     series=\normalfont}
594 \setbeamerfont{standout}{size=\Large,%
595                                     series=\bfseries}

```

8.4.4 Title format options

`titleformat title` Controls the format of the title.

```

596 \pgfkeys{
597   /metropolis/font/titleformat title/.cd,
598   .is choice,
599   regular/.code={%
600     \let\metropolis@titleformat\@empty%
601     \setbeamerfont{title}{shape=\normalfont}%
602   },
603   smallcaps/.code={%
604     \let\metropolis@titleformat\@empty%
605     \setbeamerfont{title}{shape=\scshape}%
606   },
607   allsmallcaps/.code={%
608     \let\metropolis@titleformat\lowercase%
609     \setbeamerfont{title}{shape=\scshape}%
610     \PackageWarning{beamerthememetropolis}{%
611       Be aware that titleformat title=allsmallcaps can lead to problems%
612     }
613   },
614   allcaps/.code={%
615     \let\metropolis@titleformat\uppercase%
616     \setbeamerfont{title}{shape=\normalfont}
617     \PackageWarning{beamerthememetropolis}{%
618       Be aware that titleformat title=allcaps can lead to problems%
619     }
620   },
621 }

```

`titleformat subtitle` Control the format of the subtitle.

```
622 \pgfkeys{
623   /metropolis/font/titleformat subtitle/.cd,
624   .is choice,
625   regular/.code={%
626     \let\metropolis@subtitleformat\@empty%
627     \setbeamerfont{subtitle}{shape=\normalfont}%
628   },
629   smallcaps/.code={%
630     \let\metropolis@subtitleformat\@empty%
631     \setbeamerfont{subtitle}{shape=\scshape}%
632   },
633   allsmallcaps/.code={%
634     \let\metropolis@subtitleformat\lowercase%
635     \setbeamerfont{subtitle}{shape=\scshape}%
636     \PackageWarning{beamerthememetropolis}{%
637       Be aware that titleformat subtitle=allsmallcaps can lead to problems%
638     }
639   },
640   allcaps/.code={%
641     \let\metropolis@subtitleformat\uppercase%
642     \setbeamerfont{subtitle}{shape=\normalfont}%
643     \PackageWarning{beamerthememetropolis}{%
644       Be aware that titleformat subtitle=allcaps can lead to problems%
645     }
646   },
647 }
```

`titleformat section` Controls the format of the section title.

```
648 \pgfkeys{
649   /metropolis/font/titleformat section/.cd,
650   .is choice,
651   regular/.code={%
652     \let\metropolis@sectiontitleformat\@empty%
653     \setbeamerfont{section title}{shape=\normalfont}%
654   },
655   smallcaps/.code={%
656     \let\metropolis@sectiontitleformat\@empty%
657     \setbeamerfont{section title}{shape=\scshape}%
658   },
659 }
```

```

658 },
659 allsmallcaps/.code={%
660   \let\metropolis@sectiontitleformat\MakeLowercase%
661   \setbeamerfont{section title}{shape=\scshape}%
662   \PackageWarning{beamerthememetropolis}{%
663     Be aware that titleformat section=allsmallcaps can lead to problems%
664   }
665 },
666 allcaps/.code={%
667   \let\metropolis@sectiontitleformat\MakeUppercase%
668   \setbeamerfont{section title}{shape=\normalfont}%
669   \PackageWarning{beamerthememetropolis}{%
670     Be aware that titleformat section=allcaps can lead to problems%
671   }
672 },
673 }

```

frametitleformat Control the format of the frame title.

```

674 \pgfkeys{
675   /metropolis/font/titleformat frame/.cd,
676   .is choice,
677   regular/.code={%
678     \let\metropolis@frametitleformat\@empty%
679     \setbeamerfont{frametitle}{shape=\normalfont}%
680   },
681   smallcaps/.code={%
682     \let\metropolis@frametitleformat\@empty%
683     \setbeamerfont{frametitle}{shape=\scshape}%
684   },
685   allsmallcaps/.code={%
686     \let\metropolis@frametitleformat\MakeLowercase%
687     \setbeamerfont{frametitle}{shape=\scshape}%
688     \PackageWarning{beamerthememetropolis}{%
689       Be aware that titleformat frame=allsmallcaps can lead to problems%
690     }
691   },
692   allcaps/.code={%
693     \let\metropolis@frametitleformat\MakeUppercase%
694     \setbeamerfont{frametitle}{shape=\normalfont}

```

```

695     \PackageWarning{beamerthememetropolis}{%
696         Be aware that titleformat frame=allcaps can lead to problems%
697     }
698 },
699 }

```

`titleformat aliases` Allows `titleformat title` et al. to be used in the `\usetheme` declaration, where L^AT_EX automatically removes all spaces.

```

700 \pgfkeys{
701   /metropolis/font/.cd,
702   titleformattitle/.code=\pgfkeysalso{titleformat title=#1},
703   titleformatsubtitle/.code=\pgfkeysalso{titleformat subtitle=#1},
704   titleformatsection/.code=\pgfkeysalso{titleformat section=#1},
705   titleformatframe/.code=\pgfkeysalso{titleformat frame=#1},
706 }

```

`\metropolis@font@setdefaults` Sets default values for font theme options.

```

707 \newcommand{\metropolis@font@setdefaults}{
708   \pgfkeys{/metropolis/font/.cd,
709     titleformat title=regular,
710     titleformat subtitle=regular,
711     titleformat section=regular,
712     titleformat frame=regular,
713   }
714 }

```

We first define hooks to change the case format of the titles.

```

715 \def\metropolis@titleformat#1{#1}
716 \def\metropolis@subtitleformat#1{#1}
717 \def\metropolis@sectiontitleformat#1{#1}
718 \def\metropolis@frametitleformat#1{#1}

```

To make the uppercase and lowercase macros work in the title, subtitle, etc., we have to patch the appropriate `beamer` commands that set their values. This solution was suggested by Enrico Gregorio in an answer to [this StackExchange question](#).

```

719 \patchcmd{\beamer@title}%

```



```

720 {\def\inserttitle{#2}}%
721 {\def\inserttitle{\metropolis@titleformat{#2}}}%
722 {}%
723 {\PackageError{beamerfontthememetropolis}{Patching title failed}\@ehc}
724 \patchcmd{\beamer@subtitle}%
725 {\def\insertsubtitle{#2}}%
726 {\def\insertsubtitle{\metropolis@subtitleformat{#2}}}%
727 {}%
728 {\PackageError{beamerfontthememetropolis}{Patching subtitle failed}\@ehc}
729 \patchcmd{\sectionentry}
730 {\def\insertsectionhead{#2}}
731 {\def\insertsectionhead{\metropolis@sectiontitleformat{#2}}}
732 {}
733 {\PackageError{beamerfontthememetropolis}{Patching section title failed}\@ehc}
734 \@tempswafalse
735 \patchcmd{\beamer@section}
736 {\edef\insertsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{\unexpanded{#1}}}}
737 {\edef\insertsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{%
738 \noexpand\metropolis@sectiontitleformat{\unexpanded{#1}}}}
739 {\@tempswatrue}
740 {}
741 \patchcmd{\beamer@section}
742 {\def\insertsectionhead{\hyperlink{Navigation\the\c@page}{#1}}}
743 {\def\insertsectionhead{\hyperlink{Navigation\the\c@page}{%
744 \metropolis@sectiontitleformat{#1}}}}
745 {\@tempswatrue}
746 {}
747 \patchcmd{\beamer@section}
748 {\protected@edef\insertsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{#1}}}
749 {\protected@edef\insertsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{%
750 \noexpand\metropolis@sectiontitleformat{#1}}}}
751 {\@tempswatrue}
752 {}
753 \if@tempswa\else
754 \PackageError{beamerfontthememetropolis}{Patching section title failed}\@ehc
755 \fi
756 \@tempswafalse
757 \patchcmd{\beamer@subsection}
758 {\edef\insertsubsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{\unexpanded{#1}}}}
759 {\edef\insertsubsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{%

```

```

760 \noexpand\metropolis@sectiontitleformat{\unexpanded{#1}}}}
761 {\@tempswatrue}
762 {}
763 \patchcmd{\beamer@subsection}
764 {\def\insertsubsectionhead{\hyperlink{Navigation\the\c@page}{#1}}}
765 {\def\insertsubsectionhead{\hyperlink{Navigation\the\c@page}{%
766 \metropolis@sectiontitleformat{#1}}}}
767 {\@tempswatrue}
768 {}
769 \patchcmd{\beamer@subsection}
770 {\protected@edef\insertsubsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{#1}}}
771 {\protected@edef\insertsubsectionhead{\noexpand\hyperlink{Navigation\the\c@page}{%
772 \noexpand\metropolis@sectiontitleformat{#1}}}}
773 {\@tempswatrue}
774 {}
775 \if@tempswa\else
776 \PackageError{beamerfontthememetropolis}{Patching section title failed}\@ehc
777 \fi

```

Similarly, to make the `\MakeLowercase` and `\MakeUppercase` macros work in the frame title we have to patch `\beamer@@frametitle`.

```

778 \patchcmd{\beamer@@frametitle}
779 {{%
780 \gdef\insertframetitle{{#2\ifnum\beamer@autobreakcount>0\relax{}\space%
781 \usebeamertemplate*{frametitle continuation}\fi}}%
782 \gdef\beamer@frametitle{#2}%
783 \gdef\beamer@shortframetitle{#1}%
784 }}
785 {{%
786 \gdef\insertframetitle{{\metropolis@frametitleformat{#2}\ifnum%
787 \beamer@autobreakcount>0\relax{}\space%
788 \usebeamertemplate*{frametitle continuation}\fi}}%
789 \gdef\beamer@frametitle{#2}%
790 \gdef\beamer@shortframetitle{#1}%
791 }}
792 {}
793 {\PackageError{beamerfontthememetropolis}{Patching frame title failed}\@ehc}

```

8.4.5 Process package options

```
794 \metropolis@font@setdefaults
795 \ProcessPgfPackageOptions{/metropolis/font}
```

8.5 metropolis color theme

8.5.1 Package dependencies

```
796 \RequirePackage{pgfopts}
```

8.5.2 Options

block Optionally adds a light grey background to block environments like **theorem** and **example**.

```
797 \pgfkeys{
798   /metropolis/color/block/.cd,
799   .is choice,
800   transparent/.code=\metropolis@block@transparent,
801   fill/.code=\metropolis@block@fill,
802 }
```

colors Provides the option to have a dark background and light foreground instead of the reverse.

```
803 \pgfkeys{
804   /metropolis/color/background/.cd,
805   .is choice,
806   dark/.code=\metropolis@colors@dark,
807   light/.code=\metropolis@colors@light,
808 }
```

\metropolis@color@setdefaults Sets default values for color theme options.

```
809 \newcommand{\metropolis@color@setdefaults}{
810   \pgfkeys{/metropolis/color/.cd,
811     background=light,
812     block=transparent,
813   }
814 }
```

8.5.3 Base colors

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

8.5.4 Base styles

All colors in **metropolis** are derived from the definitions of **normal text**, **alerted text**, and **example text**.

```
819 \newcommand{\metropolis@colors@dark}{  
820   \setbeamercolor{normal text}{%  
821     fg=black!2,  
822     bg=mDarkTeal  
823   }  
824   \usebeamercolor[fg]{normal text}  
825 }  
826 \newcommand{\metropolis@colors@light}{  
827   \setbeamercolor{normal text}{%  
828     fg=mDarkTeal,  
829     bg=black!2  
830   }  
831 }  
832 \setbeamercolor{alerted text}{%  
833   fg=mLightBrown  
834 }  
835 \setbeamercolor{example text}{%  
836   fg=mLightGreen  
837 }
```

8.5.5 Derived colors

The titles and structural elements (e.g. **itemize** bullets) are set in the same color as **normal text**. This would ideally be 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`.

```
838 \setbeamercolor{titlelike}{use=normal text, parent=normal text}
```

```

839 \setbeamercolor{author}{use=normal text, parent=normal text}
840 \setbeamercolor{date}{use=normal text, parent=normal text}
841 \setbeamercolor{institute}{use=normal text, parent=normal text}
842 \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. **metropolis** uses it for frame titles and slides.

```

843 \setbeamercolor{palette primary}{%
844   use=normal text,
845   fg=normal text.bg,
846   bg=normal text.fg
847 }
848 \setbeamercolor{frametitle}{%
849   use=palette primary,
850   parent=palette primary
851 }

```

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`.

```

852 \setbeamercolor{progress bar}{%
853   use=alerted text,
854   fg=alerted text.fg,
855   bg=alerted text.fg!50!black!30
856 }
857 \setbeamercolor{title separator}{
858   use=progress bar,
859   parent=progress bar
860 }
861 \setbeamercolor{progress bar in head/foot}{%
862   use=progress bar,
863   parent=progress bar
864 }
865 \setbeamercolor{progress bar in section page}{
866   use=progress bar,
867   parent=progress bar
868 }

```

Block environments such as `theorem` and `example` have no background color by default. The option `block=fill` sets a background color based on the background and foreground of `normal text`. The option `block=transparent` reverts the block environments to an empty background, which can be useful if changing colors mid-presentation.

```

869 \newcommand{\metropolis@block@transparent}{
870   \setbeamercolor{block title}{%
871     use=normal text,
872     fg=normal text.fg,
873     bg=
874   }
875   \setbeamercolor{block body}{
876     bg=
877   }
878 }
879 \newcommand{\metropolis@block@fill}{
880   \setbeamercolor{block title}{%
881     use=normal text,
882     fg=normal text.fg,
883     bg=normal text.bg!80!fg
884   }
885   \setbeamercolor{block body}{
886     use={block title, normal text},
887     bg=block title.bg!50!normal text.bg
888   }
889 }
890 \setbeamercolor{block title alerted}{%
891   use={block title, alerted text},
892   bg=block title.bg,
893   fg=alerted text.fg
894 }
895 \setbeamercolor{block title example}{%
896   use={block title, example text},
897   bg=block title.bg,
898   fg=example text.fg
899 }
900 \setbeamercolor{block body alerted}{use=block body, parent=block body}
901 \setbeamercolor{block body example}{use=block body, parent=block body}

```

Footnotes

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

We also reset the bibliography colors in order to pick up the surrounding colors at the time of use. This prevents us having to set the correct color in normal and standout mode.

```
904 \setbeamercolor{bibliography entry author}{fg=, bg=}  
905 \setbeamercolor{bibliography entry title}{fg=, bg=}  
906 \setbeamercolor{bibliography entry location}{fg=, bg=}  
907 \setbeamercolor{bibliography entry note}{fg=, bg=}
```

8.5.6 Process package options

```
908 \metropolis@color@setdefaults  
909 \ProcessPgfPackageOptions{/metropolis/color}  
  
910 \mode<all>
```

8.6 Tol pgfplots theme

Paul Tol’s 12-color palette¹ is as follows:

```
911 \definecolor{TolDarkPurple}{HTML}{332288}  
912 \definecolor{TolDarkBlue}{HTML}{6699CC}  
913 \definecolor{TolLightBlue}{HTML}{88CCEE}  
914 \definecolor{TolLightGreen}{HTML}{44AA99}  
915 \definecolor{TolDarkGreen}{HTML}{117733}  
916 \definecolor{TolDarkBrown}{HTML}{999933}  
917 \definecolor{TolLightBrown}{HTML}{DDCC77}  
918 \definecolor{TolDarkRed}{HTML}{661100}  
919 \definecolor{TolLightRed}{HTML}{CC6677}  
920 \definecolor{TolLightPink}{HTML}{AA4466}  
921 \definecolor{TolDarkPink}{HTML}{882255}  
922 \definecolor{TolLightPurple}{HTML}{AA4499}
```

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

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

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

```
923 \pgfplotscreateplotcyclelist{mbarplot cycle}{%
924 {draw=TolDarkBlue, fill=TolDarkBlue!70},
925 {draw=TolLightBrown, fill=TolLightBrown!70},
926 {draw=TolLightGreen, fill=TolLightGreen!70},
927 {draw=TolDarkPink, fill=TolDarkPink!70},
928 {draw=TolDarkPurple, fill=TolDarkPurple!70},
929 {draw=TolDarkRed, fill=TolDarkRed!70},
930 {draw=TolDarkBrown, fill=TolDarkBrown!70},
931 {draw=TolLightRed, fill=TolLightRed!70},
932 {draw=TolLightPink, fill=TolLightPink!70},
933 {draw=TolLightPurple, fill=TolLightPurple!70},
934 {draw=TolLightBlue, fill=TolLightBlue!70},
935 {draw=TolDarkGreen, fill=TolDarkGreen!70},
936 }
```

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

```
937 \pgfplotscreateplotcyclelist{mlineplot cycle}{%
938 {TolDarkBlue, mark=*, mark size=1.5pt},
939 {TolLightBrown, mark=square*, mark size=1.3pt},
940 {TolLightGreen, mark=triangle*, mark size=1.5pt},
941 {TolDarkBrown, mark=diamond*, mark size=1.5pt},
942 }
```

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.

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

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

```
945 mlineplot/.style={
946 mbaseplot,
947 xmajorgrids=true,
948 ymajorgrids=true,
949 major grid style={dotted},
```



```

950     axis x line=bottom,
951     axis y line=left,
952     legend style={
953         cells={anchor=west},
954         draw=none
955     },
956     cycle list name=mlineplot cycle,
957 },

```

`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`.

```

958 mbarplot base/.style={
959     mbaseplot,
960     bar width=6pt,
961     axis y line*=none,
962 },
963 mbarplot/.style={
964     mbarplot base,
965     ybar,
966     xmajorgrids=false,
967     ymajorgrids=true,
968     area legend,
969     legend image code/.code={%
970         \draw[#1] (0cm,-0.1cm) rectangle (0.15cm,0.1cm);
971     },
972     cycle list name=mbarplot cycle,
973 },
974 horizontal mbarplot/.style={
975     mbarplot base,
976     xmajorgrids=true,
977     ymajorgrids=false,
978     xbar stacked,
979     area legend,
980     legend image code/.code={%
981         \draw[#1] (0cm,-0.1cm) rectangle (0.15cm,0.1cm);
982     },
983     cycle list name=mbarplot cycle,
984 },

```

`mbaseplot` Adjusts the appearance of the axes in a PGF chart.

```
985 mbaseplot/.style={
986     legend style={
987         draw=none,
988         fill=none,
989         cells={anchor=west},
990     },
991     x tick label style={
992         font=\footnotesize
993     },
994     y tick label style={
995         font=\footnotesize
996     },
997     legend style={
998         font=\footnotesize
999     },
1000     major grid style={
1001         dotted,
1002     },
1003     axis x line*=bottom,
1004 },
1005 disable thousands separator/.style={
1006     /pgf/number format/.cd,
1007     1000 sep={}
1008 },
1009 }
```