

codebox: programming code box

Nan Geng

nangeng@nwafu.edu.cn

2021/12/27 v1.0.1^{*†}

Abstract

codebox is a tcolorbox-based package developed with L^AT_EX3, which provides environments codebox and codeview, and macros \codefile and \cvfile for typesetting programming source code box.

The environments create codebox with its body and macros is used to read in the source code file and output is in the codebox.

The starred environments and macros are also provided to get codebox with comments at the bottom of box.

All codebox style can be set by \codeset macro or environment's and macro's key-value [*options*].

Contents

1	introduction	1	3.3	title prefix	4
2	interface	2	3.4	code highlight style	5
2.1	codebox and codebox* environments	2	3.5	code fontsize	5
2.2	\codefile and \codefile* macros	2	3.6	comment contents	5
2.3	codeview and codeview* environments	3	3.7	comment format	5
2.4	\cvfile and \cvfile* macros	4	3.8	code baseline stretch	5
3	Options	4	3.9	separation between line number and code	5
3.1	code engine	4	4	Examples	5
3.2	language	4	4.1	Java code	6
			4.2	Python code	6
			4.3	listings engine	7

1 introduction

codebox is a L^AT_EX3 package for typesetting programming source code box.

Both codebox and codeview environment are provided with environment body. At the same time, both \codefile and \cvfile macros are created for reading source code file.

The starred environments (codebox* and codeview*) and macros (\codefile* and \cvfile*) are also provided to get codebox with comments at the bottom of box.

^{*}<https://github.com/registor/codebox>

[†]https://gitee.com/nwafu_nan/codebox

2 interface

2.1 codebox and codebox* environments

codebox	\begin{codebox}[\langle options \rangle]{\langle codebox title \rangle}
codebox*
	\end{codebox}

New: 2021-12-25	\begin{codebox*}[\langle options \rangle]{\langle codebox title \rangle}
Updated: 2021-12-25
	\end{codebox*}

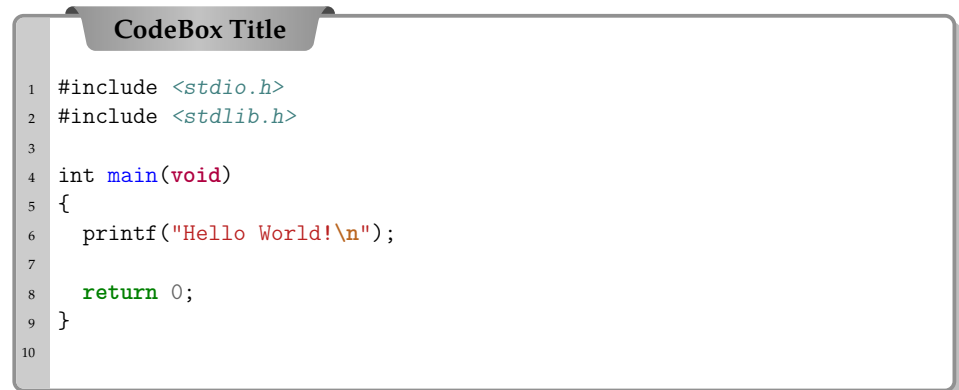
Typesetting codebox with environment body. You can set the title of the codebox with $\{\langle codebox title \rangle\}$.

The appearance of the codebox is set by key-value in $[\langle options \rangle]$.

The starred environment `codebox*` is used to add comments at the bottom of the codebox, note that this needs to be done with $\langle comments \rangle = \langle texts \rangle$ in $[\langle options \rangle]$.

Of course the key-value $[\langle options \rangle]$ can also be set via the comma-separated key-value list of the `\codeset` macro.

```
1 \centering
2 \begin{codebox}{CodeBox Title}
3   \include <stdio.h>
4   \include <stdlib.h>
5
6   int main(void)
7   {
8       printf("Hello World!\n");
9
10      return 0;
11  }
12 \end{codebox}
```



2.2 \codefile and \codefile* macros

\codefile	\codefile [\langle options \rangle] {\langle codebox title \rangle} {\langle code file \rangle}
\codefile*	\codefile* [\langle options \rangle] {\langle codebox title \rangle} {\langle code file \rangle}

New: 2021-12-25
Updated: 2021-12-25

Typesetting codebox from a source code file. You can set the title of the codebox with $\{\langle codebox title \rangle\}$.

The appearance of the codebox is set by key-value in $[\langle options \rangle]$.

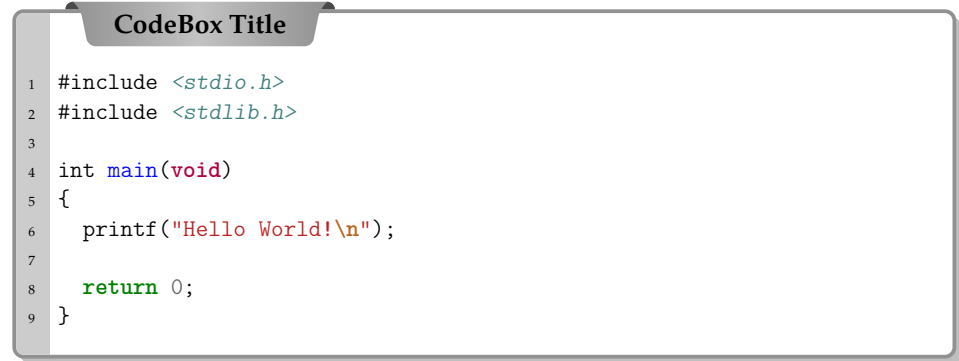
The starred environment `\codefile*` is used to add comments at the bottom of the codebox, note that this needs to be done with $\langle comments \rangle = \langle texts \rangle$ in $[\langle options \rangle]$.

Of course the key-value $[\langle options \rangle]$ can also be set via the comma-separated key-value list of the `\codeset` macro.

```

1 \centering
2 \codefile{CodeBox Title}{test.c}

```



2.3 codeview and codeview* environments

codeview
codeview*

New: 2021-12-26

Updated: 2021-12-26

```

\begin{codeview}[\langle options \rangle]{\langle codeview title \rangle}
.....
\end{codeview}
\begin{codeview*}[\langle options \rangle]{\langle codeview title \rangle}
.....
\end{codeview*}

```

Typesetting code viewer with environment body. You can set the title of the code viewer with $\{\langle codeview title \rangle\}$.

The appearance of the code viewer is set by key-value in $[\langle options \rangle]$.

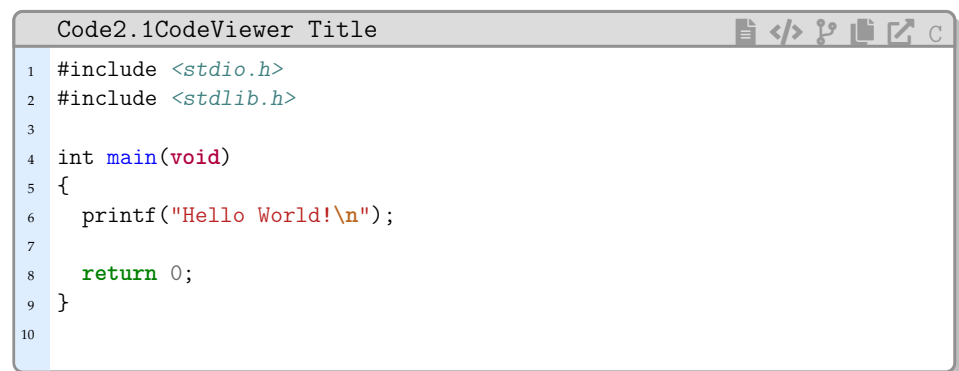
The starred environment `codeview*` is used to add comments at the bottom of the codebox, note that this needs to be done with $\langle comments \rangle = \langle texts \rangle$ in $[\langle options \rangle]$.

Of course the key-value $[\langle options \rangle]$ can also be set via the comma-separated key-value list of the `\codeset` macro.

```

1 \centering
2 \begin{codeview}{CodeViewer Title}
3   #include <stdio.h>
4   #include <stdlib.h>
5
6   int main(void)
7   {
8     printf("Hello World!\n");
9
10    return 0;
11  }
12 \end{codeview}

```



2.4 \cvfile and \cvfile* macros

\cvfile	\cvfile [<i>options</i>] { <i>codeview title</i> } { <i>code file</i> }
\cvfile*	\cvfile* [<i>options</i>] { <i>codeview title</i> } { <i>code file</i> }

New: 2021-12-26
Updated: 2021-12-26

Typesetting code viewer from a source code file. You can set the title of the code viewer with {*codeview title*}.

The appearance of the code viewer is set by key-value in [*options*].

The starred environment \vcfile* is used to add comments at the bottom of the codebox, note that this needs to be done with <comments> = <texts> in [*options*].

Of course the key-value [*options*] can also be set via the comma-separated key-value list of the \codeset macro.

```
1 \centering
2 \cvfile*[comments=this is a simple C code]{CodeViewer Title}{test.c}
```



3 Options

The codebox package provides a number of options to set the style of the codebox. The following options can be set with \codeset macro. Also, these options can be set with the all environment's or command's [*options*].

3.1 code engine

minted	minted = { <i>true false</i> }	Init = true
New: 2021-12-26 Updated: 2021-12-26	minted is used to set code highlight engine, if it is true then the minted package is used, if it is false then the listings package is used. The default is true .	

3.2 language

lang	lang = { <i>source code language</i> }	Init = C
New: 2021-12-26 Updated: 2021-12-26	lang is used to set source code language. The default is C .	

3.3 title prefix

pretitle	pretitle = { <i>title prefix</i> }	Init = Code
New: 2021-12-26 Updated: 2021-12-26	pretitle is used to set prefix of code counter. The default is Code .	

3.4 code highlight style

codestyle	<code>codestyle = {\<highlight style>}</code>	Init = codeblocks
New: 2021-12-26 Updated: 2021-12-26	<code>codestyle</code> is used to set code highlight style, valid only for the minted engine. The default is codeblocks .	

3.5 code fontsize

codesize	<code>codesize = {\<fontsize macro>}</code>	Init = \small
New: 2021-12-26 Updated: 2021-12-26	<code>codesize</code> is used to set code fontsize, valid only for minted engine. The default is \small .	

3.6 comment contents

comments	<code>comments = {\<texts>}</code>	Init = nothing
New: 2021-12-26 Updated: 2021-12-26	<code>comments</code> is used to set comment contents. The default is nothing .	

3.7 comment format

commentf	<code>commentf = {\<format macros>}</code>	Init = \small\sffamily
New: 2021-12-26 Updated: 2021-12-26	<code>commentf</code> is used to set comment format at codebox bottom. The default is \small\sffamily .	

3.8 code baseline stretch

codestretch	<code>codestretch = {\<float number>}</code>	Init = 1.0
New: 2021-12-26 Updated: 2021-12-26	<code>codestretch</code> is used to set code baseline stretch, valid only for minted engine. The default is 1.0 .	

3.9 seperation between line number and code

linenumsep	<code>linenumsep = {\<float number>}</code>	Init = 3.0
New: 2021-12-26 Updated: 2021-12-26	<code>linenumsep</code> is used to set the seperation between line number and code, valid only for minted engine. Note the unit is mm. The default is 3.0 .	

4 Examples

The codebox package can be used in situations where the highlight programming source code needs to be typeset to avoid the use of screenshots. Code box can be with or without underline comments.

4.1 Java code

The language can be set with \codeset macro.

```
1 \centering
2 \codeset{lang=java}
3 \codefile{Java CodeBox}{hellojava.java}
```

Java CodeBox

```
1 public class HelloWorld {
2     public static void main(String[] args){
3         System.out.println("Hello World!");
4     }
5 }
```

4.2 Python code

The language can be set with options.

```
1 \centering
2 \cvfile[lang=python]{Python CodeBox}{hellopy.py}
```

Code4.3Python CodeBox

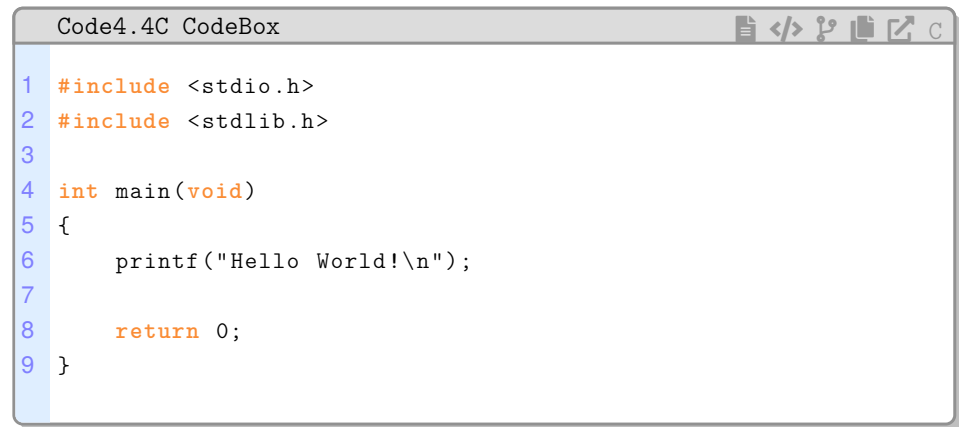
     PYTHON

```
1 import tensorflow as tf
2 import numpy as np
3 import os
4 os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
5
6 # Create 100 phony x, y data points in Numpy, y = x * 0.1 + 0.3
7 x_data = np.random.random(100).astype("float32")
8 y_data = x_data * 0.1 + 0.3
9
10 # Try to find values for W and b that compute y_data = W * x_data + b
11 W = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
12 b = tf.Variable(tf.zeros([1]))
13 y = W * x_data + b
14
15 # Minimize the mean squared errors.
16 loss = tf.reduce_mean(tf.square(y - y_data))
17 optimizer = tf.train.GradientDescentOptimizer(0.5)
18 train = optimizer.minimize(loss)
19
20 # Before starting, initialize the variables. We will 'run' this first
21 init = tf.global_variables_initializer()
22
23 # Launch the graph.
24 sess = tf.Session()
25 sess.run(init)
26
27 # Fit the line.
28 for step in range(201):
29     sess.run(train)
30     if step % 20 == 0:
31         print(step, sess.run(W), sess.run(b))
32
```

4.3 listings engine

listings engine can be set with `<minted>=<false>`.

```
1 \centering
2 \cvfile[minted=false,lang=c]{C CodeBox}{test.c}
```



```
Code4.4C CodeBox
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(void)
5 {
6     printf("Hello World!\n");
7
8     return 0;
9 }
```