

Codehigh: Highlight Codes and Demos with I3RegEx and LPeg

Jianrui Lyu (tolvjr@163.com)
<https://github.com/lvjrr/codehigh>

Version 2022D (2022-02-07)

Contents

1	Package Interfaces	2
1.1	Introduction	2
1.2	Highlighting Code	2
1.3	Highlighting Demo	3
1.4	Highlighting File	4
1.5	Customization	4
2	The Source Code	5
2.1	Variables and Functions	5
2.2	Set CodeHigh Options	6
2.3	CodeHigh Environments and Commands	6
2.4	Typeset CodeHigh Code	8
2.5	Typeset CodeHigh Demo	10
2.6	Add CodeHigh Languages	12
2.7	Add CodeHigh Themes	13
2.8	Parse and Highlight Code	14
2.9	Don't Highlight Code	16

Chapter 1

Package Interfaces

1.1 Introduction

Codehigh package uses `l3regex`¹ package in L^AT_EX3 Programming Layer to parse and highlight source codes and demos. It is more powerful than `listings` package, and more easy to use than `minted` package. But it is slower than both of them. Therefore in LuaTeX the package provides another way to highlight code: using LPeg². LPeg is much more powerful and faster than `l3regex`.

1.2 Highlighting Code

There are several predefined languages: `latex`, `latex/latex2`, `latex/latex3`, `latex/math` and `latex/table`. The following example is typeset by `codehigh` environment with default option `language=latex`.

```
\documentclass{article}
\usepackage[a4paper,margin=2cm]{geometry}
\usepackage{codehigh}
\usepackage{hyperref}
\newcommand*{\myversion}{2021C}
\newcommand*{\mydate}{Version \myversion\ (\the\year-\mylpad\month-\mylpad\day)}
\newcommand*{\mylpad}[1]{\ifnum#1<10 0\the#1\else\the#1\fi}
\setlength{\abc}{1}
\begin{document}
% some comment
\section{Section Name}
\subsection*{Suction Name}
Math  $a+b$ .
\end{document}
```

The following example is typeset by `codehigh` environment with option `language=latex/latex2`.

```
\def\abcd#1#2{
% some comment
\unskip
\setlength{\parindent}{0pt}%
\setlength{\parskip}{0pt}%
\setcounter{choice}{0}%
\let\item=\my@item@temp
\settowidth{\my@item@len}{\vbox{\halign{##1\hfil\cr\BODY\crrc}}}%
\setcounter{choice}{0}%
}
```

¹<https://www.ctan.org/pkg/l3regex>

²<http://www.inf.puc-rio.br/~roberto/lpeg/>

This language is for highlighting LaTeX2 classes and packages. It highlights private commands and public commands with different colors.

The following example is typeset by `codehigh` environment with option `language=latex/latex3`.

```
\cs_new_protected:Npn \__codehigh_typeset_demo:
{
  \__codehigh_build_code:
  \__codehigh_build_demo:
  \dim_set:Nn \l_tmpa_dim { \box_wd:N \g__codehigh_code_box }
  \dim_set:Nn \l_tmpb_dim { \box_wd:N \g__codehigh_demo_box }
  \par\addvspace{0.5em}\noindent
  % more code
}
```

This language is for highlighting LaTeX3 classes and packages. It highlights private commands/variables and public commands/variables with different colors.

The following example is typeset by `codehigh` environment with option `language=latex/math`.

```
\begin{align}
\pi\left[\frac{13z^3}{\right]}\sin(2x+1)_0^4 = \frac{64}{3}\pi
\end{align}
```

The following example is typeset by `codehigh` environment with option `language=latex/table`.

```
\begin{tabular}[b]{|lc|rl|}
\hline
One & & Two & & Three & \\\
%\hline
Four & & Five & & Six & \\\
\hline%\hline\hline
Seven & & Eight & & Nine & \\\
\hline
\end{tabular}
```

1.3 Highlighting Demo

The followings are typeset by `demohigh` environment with option `language=latex/table`.

```
\begin{tabular}{lccr}
\hline
Alpha & & Beta & & Gamma & & Delta & \\\
\hline
Epsilon & & Zeta & & Eta & & Theta & \\\
\hline
Iota & & Kappa & & Lambda & & Mu & \\\
\hline
\end{tabular}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

```
\begin{tabular}{llccrr}
\hline
Alpha & & Beta & & Gamma & & Delta & & Epsilon & & Zeta & \\\
\hline
Eta & & Theta & & Iota & & Kappa & & Lambda & & Mu & \\\
\hline
\end{tabular}
```

Alpha	Beta	Gamma	Delta	Epsilon	Zeta
Eta	Theta	Iota	Kappa	Lambda	Mu

Note that `demohigh` environment will measure the width of source lines. When it is too large, the result will be put below.

1.4 Highlighting File

Using `\dochighinput` command, you can input and highlight some file. The last chapter of this manual is typeset with the following code line:

```
\dochighinput[language=latex/latex3]{codehigh.sty}
```

In reading an input file, lines starting with `%%` will be omitted, and lines starting with `%%>` will be extracted and typeset as normal text.

1.5 Customization

The following example changes default background colors with `\CodeHigh` command:

```
\CodeHigh{language=latex/table,style/main=yellow9,style/code=red9,style/demo=azure9}
```

Note that `codehigh` package will load `ninecolors`³ package for proper color contrast.

```
\begin{tabular}{lccr}
\hline
Alpha    & Beta   & Gamma  & Delta  \\\
\hline
Epsilon  & Zeta   & Eta    & Theta  \\\
\hline
Iota     & Kappa  & Lambda & Mu     \\\
\hline
\end{tabular}
```

Alpha	Beta	Gamma	Delta
Epsilon	Zeta	Eta	Theta
Iota	Kappa	Lambda	Mu

To modify or add languages and themes, please read the source files `codehigh.sty` and `codehigh.lua` for reference.

³<https://www.ctan.org/pkg/ninecolors>

Chapter 2

The Source Code

2.1 Variables and Functions

```
\NeedsTeXFormat{LaTeX2e}
\RequirePackage{expl3}
\ProvidesExplPackage{codehigh}{2022-02-07}{2022D}
  {Highlight codes and demos with l3regex and lpeg}

%\RequirePackage{xparse}
\RequirePackage{l3benchmark}
\RequirePackage{catchfile}
\RequirePackage{xcolor}
\RequirePackage{ninecolors}
\RequirePackage{varwidth}
\ifluatex
  \RequirePackage{luatexbase}
\fi

\int_new:N \l__codehigh_a_int
\int_new:N \l__codehigh_b_int
\tl_new:N \l__codehigh_a_tl
\tl_new:N \l__codehigh_b_tl
\tl_new:N \l__codehigh_c_tl
\tl_new:N \l__codehigh_d_tl
\tl_new:N \l__codehigh_m_tl

\cs_generate_variant:Nn \regex_set:Nn {cn}
\cs_generate_variant:Nn \seq_set_split:Nnn {NVV}
\cs_generate_variant:Nn \str_remove_once:Nn {NV}
\cs_generate_variant:Nn \tl_set_rescan:Nnn {NnV}

\prg_generate_conditional_variant:Nnn \str_if_eq:nn {en} {T, TF}
\prg_generate_conditional_variant:Nnn \regex_extract_once:NnN {NVN, cVN} {T, TF}
\prg_generate_conditional_variant:Nnn \regex_split:NnN {cVN} {T, TF}

\group_begin:
```

```

\obeylines
\tl_gset:Nn \g__codehigh_eol_tl {\^M}
\tl_gset:Nn \g__codehigh_eol_eol_tl {\^M\^M}
\group_end:

```

2.2 Set CodeHigh Options

```

\bool_new:N \l__codehigh_lite_bool
\bool_new:N \l__codehigh_long_bool
\bool_new:N \l__codehigh_demo_bool

\NewDocumentCommand \CodeHigh {0{} m}
{
  \keys_set:nn {codehigh} {#2}
}

\keys_define:nn {codehigh}
{
  lite .bool_set:N = \l__codehigh_lite_bool,
  long .bool_set:N = \l__codehigh_long_bool,
  demo .bool_set:N = \l__codehigh_demo_bool,
}

```

2.3 CodeHigh Environments and Commands

```

\NewDocumentCommand \NewCodeHighEnv {mm}
{
  \NewDocumentEnvironment {#1} {0{}}
  {
    \keys_set:nn {codehigh} {#2, ##1}
    \bgroup
    \__codehigh_do_specials:
    \begin{codehigh@aux}
  }
  {
    \end{codehigh@aux}
    \egroup
    \__codehigh_typeset:
  }
}

\cs_new_protected:Npn \__codehigh_do_specials:
{
  \obeylines
  \obeyspaces
}

```

```

\catcode `\\% = 12
\catcode `\\^ = 12
}

\tl_new:N \g__codehigh_code_tl

\NewDocumentEnvironment {codehigh@aux} {+b}
{
  \tl_gset:Nn \g__codehigh_code_tl { #1 }
  %\tl_log:N \g__codehigh_code_tl
}
{ }

\cs_new_protected:Npn \__codehigh_typeset:
{
  \bool_if:NTF \l__codehigh_demo_bool
    {\__codehigh_typeset_demo:} {\__codehigh_typeset_code:}
}

\NewCodeHighEnv {codehigh} {}
\NewCodeHighEnv {demohigh} {demo}

\tl_new:N \l__codehigh_input_tl
\seq_new:N \l__codehigh_input_seq

\NewDocumentCommand \NewCodeHighInput {mm}
{
  \NewDocumentCommand #1 {O{}}m
  {
    \group_begin:
    \keys_set:nn {codehigh} {#2, ##1}
    \CatchFileDef \l__codehigh_input_tl {##2} {\__codehigh_do_specials:}
    \__codehigh_typeset_input:N \l__codehigh_input_tl
    \group_end:
  }
}

\cs_new_protected:Npn \__codehigh_typeset_input:N #1
{
  \seq_set_split:NVV \l__codehigh_input_seq \g__codehigh_eol_eol_tl #1
  \seq_map_inline:Nn \l__codehigh_input_seq
  {
    \tl_gset:Nn \g__codehigh_code_tl {##1}
    \__codehigh_typeset_comment:N \g__codehigh_code_tl
    \tl_if_blank:VF \g__codehigh_code_tl
    {

```



```

        \_codehigh\_typeset\_code:
        \par
        \medskip
    }
}

\regex\_const:Nn \l\_codehigh\_comment\_regex { ^ \% \% ( [\%>] ) ( [^\r]+ ) [\r] }
\tl\_new:N \l\_codehigh\_comment\_tl % comment lines that need to be typeset
\bool\_new:N \l\_codehigh\_comment\_bool

%% remove lines starting with %%, and typeset lines starting with %>
\cs\_new\_protected:Npn \_codehigh\_typeset\_comment:N #1
{
    \tl\_set\_eq:NN \l\_tmpa\_tl #1
    \tl\_put\_right:NV \l\_tmpa\_tl \g\_codehigh\_eol\_tl
    \tl\_clear:N \l\_codehigh\_comment\_tl
    \bool\_set\_false:N \l\_codehigh\_comment\_bool
    \bool\_do\_until:Nn \l\_codehigh\_comment\_bool
    {
        %% Unfortunately we need both \regex\_extract\_once and \regex\_replace\_once
        \regex\_extract\_once:NVNTF \l\_codehigh\_comment\_regex \l\_tmpa\_tl \l\_tmpa\_seq
        {
            \regex\_replace\_once:NnN \l\_codehigh\_comment\_regex {} \l\_tmpa\_tl
            \str\_if\_eq:enT { \seq\_item:Nn \l\_tmpa\_seq {2} } {>}
            {
                \tl\_put\_right:Nx \l\_codehigh\_comment\_tl
                { \seq\_item:Nn \l\_tmpa\_seq {3} }
            }
        }
        { \bool\_set\_true:N \l\_codehigh\_comment\_bool }
    }
    \exp\_args:NV \scantokens \l\_codehigh\_comment\_tl
    \tl\_gset\_eq:NN #1 \l\_tmpa\_tl
}

\NewCodeHighInput \dochighinput {long}

```

2.4 Typeset CodeHigh Code

```

\dim\_new:N \l\_codehigh\_main\_boxsep\_dim

\keys\_define:nn {codehigh}
{
    boxsep .dim\_set:N = \l\_codehigh\_main\_boxsep\_dim,

```

```

    boxsep .initial:n = 3pt,
}

\box_new:N \g__codehigh_code_box

\cs_new_protected:Npn \__codehigh_typeset_code:
{
  \par\addvspace{0.5em}\noindent
  \bool_if:NTF \l__codehigh_long_bool
    {\__codehigh_typeset_code_text:} {\__codehigh_typeset_code_box:}
  \par\addvspace{0.5em}
}

\cs_new_protected:Npn \__codehigh_typeset_code_text:
{
  \__codehigh_prepare_code:N \l_tmpa_tl
  \__codehigh_get_code_text:n \l_tmpa_tl
}

\cs_new_protected:Npn \__codehigh_typeset_code_box:
{
  \__codehigh_build_code:
  \__codehigh_put_code_box:
}

\cs_new_protected:Npn \__codehigh_build_code:
{
  \__codehigh_prepare_code:N \l_tmpa_tl
  \__codehigh_get_code_box:nN \l_tmpa_tl \g__codehigh_code_box
}

\cs_new_protected:Npn \__codehigh_prepare_code:N #1
{
  \tl_set_eq:NN #1 \g__codehigh_code_tl
  \regex_replace_once:nnN {\r} {} #1
  \regex_replace_once:nnN {\r $} {} #1
  \regex_replace_all:nnN { . } { \c{string} \0 } #1
  \tl_set:Nx #1 { #1 }
}

\cs_new_protected:Npn \__codehigh_put_code_box:
{
  \setlength \fboxsep {\l__codehigh_main_boxsep_dim}
  \GetCodeHighStyle{main}
  \colorbox{codehigh@bg}
  {

```

```

\hbox_to_wd:nn {\linewidth-2\fbboxsep}
{
  \GetCodeHighStyle{code}
  \colorbox{codehigh@bg}
    {\box_use:N \g__codehigh_code_box}
}
}

%% #1: text to parse; #2: resulting box
\cs_new_protected:Npn \__codehigh_get_code_box:nN #1 #2
{
  \hbox_gset:Nn #2
  {
    \begin{varwidth}{\linewidth}
      \__codehigh_get_code_text:n {#1}
    \end{varwidth}
  }
}

\cs_new_protected:Npn \__codehigh_get_code_text:n #1
{
  \group_begin:
    \setlength \parindent {0pt}
    \linespread {1}
    \ttfamily
    \bool_if:NTF \l__codehigh_lite_bool
      {\__codehigh_parse_code_lite:N #1}
      {\__codehigh_parse_code:VN \l__codehigh_language_name_tl #1}
  \group_end:
}

```

2.5 Typeset CodeHigh Demo

```

\box_new:N \g__codehigh_demo_box

\cs_new_protected:Npn \__codehigh_typeset_demo:
{
  \__codehigh_build_code:
  \__codehigh_build_demo:
  \dim_set:Nn \l_tmpa_dim { \box_wd:N \g__codehigh_code_box }
  \dim_set:Nn \l_tmpb_dim { \box_wd:N \g__codehigh_demo_box }
  %\tl_log:x { \dim_use:N \l_tmpa_dim + \dim_use:N \l_tmpb_dim }
  \par\addvspace{0.5em}\noindent
  \setlength \fbboxsep {\l__codehigh_main_boxsep_dim}
}

```

```

\GetCodeHighStyle{main}
\colorbox{codehigh@bg}
{
  \dim_compare:nNnTF {\l_tmpa_dim + \l_tmpb_dim + 6\fbboxsep} > {\linewidth}
  {
    \vbox:n
    {
      \dim_set:Nn \hsize {\linewidth-2\fbboxsep}
      \noindent\GetCodeHighStyle{code}
      \colorbox{codehigh@bg}{\box_use:N \g__codehigh_code_box}
      \par
      \noindent\GetCodeHighStyle{demo}
      \colorbox{codehigh@bg}{\box_use:N \g__codehigh_demo_box}
    }
  }
  {
    \hbox_to_wd:nn {\linewidth-2\fbboxsep}
    {
      \GetCodeHighStyle{code}
      \colorbox{codehigh@bg}{\box_use:N \g__codehigh_code_box}
      \hfill
      \GetCodeHighStyle{demo}
      \colorbox{codehigh@bg}{\box_use:N \g__codehigh_demo_box}
    }
  }
}
\par\addvspace{0.5em}
}

\cs_new_protected:Npn \__codehigh_build_demo:
{
  \tl_set_eq:NN \l_tmpb_tl \g__codehigh_code_tl
  \tl_set_rescan:NnV \l_tmpb_tl
  {
    \catcode ` \% = 14 \relax
    \catcode ` \^M = 10 \relax
  }
  \l_tmpb_tl
  %\tl_log:N \l_tmpb_tl
  \__codehigh_get_demo_box:nN \l_tmpb_tl \g__codehigh_demo_box
}

%% #1: text to typeset; #2: resulting box
\cs_new_protected:Npn \__codehigh_get_demo_box:nN #1 #2
{
  \hbox_gset:Nn #2

```

```

{
  \dim_set:Nn \linewidth {\linewidth-4\l__codehigh_main_boxsep_dim}
  \begin{varwidth}{\linewidth}
    \setlength { \parindent } { 0pt }
    \linespread {1}
    \tl_use:N #1
  \end{varwidth}
}
}

```

2.6 Add CodeHigh Languages

```

\keys_define:nn {codehigh}
{
  language .tl_set:N = \l__codehigh_language_name_tl,
  language .initial:n = latex,
}

%% #1: language name; #2: rule type; #3: rule name; #4: rule regex
\NewDocumentCommand \AddCodeHighRule {0{latex} m m m}
{
  \int_if_exist:cF {l__codehigh_#1_rule_count_int}
  {\int_new:c {l__codehigh_#1_rule_count_int}}
  \int_incr:c {l__codehigh_#1_rule_count_int}
  \tl_set:cn
  {l__codehigh_#1_ \int_use:c {l__codehigh_#1_rule_count_int} _type_tl} {#2}
  \tl_set:cn
  {l__codehigh_#1_ \int_use:c {l__codehigh_#1_rule_count_int} _name_tl} {#3}
  \regex_set:cn
  {l__codehigh_#1_ \int_use:c {l__codehigh_#1_rule_count_int} _regex} {#4}
}

\AddCodeHighRule[latex]{1}{Package}    {\(\documentclass|usepackage)}
\AddCodeHighRule[latex]{6}{NewCommand}{\(\newcommand}
\AddCodeHighRule[latex]{3}{SetCommand}{\(\set[A-Za-z]+}
\AddCodeHighRule[latex]{4}{BeginEnd}   {\(\begin|end)}
\AddCodeHighRule[latex]{5}{Section}    {\(\<part|chapter|section|subsection)}
\AddCodeHighRule[latex]{2}{Command}    {\(\[A-Za-z]+}
\AddCodeHighRule[latex]{7}{Brace}      {\([{\}]}
\AddCodeHighRule[latex]{8}{MathMode}   {\$}
\AddCodeHighRule[latex]{9}{Comment}    {\%.*?\r}

\AddCodeHighRule[latex/math]{6}{LeftRight} {\(\left|right)}
\AddCodeHighRule[latex/math]{2}{Command}   {\(\[A-Za-z]+}
\AddCodeHighRule[latex/math]{8}{MathMode}  {\$}

```

```

\AddCodeHighRule[latex/math]{4}{Script}      {\[\_~]}
\AddCodeHighRule[latex/math]{5}{Number}      {\d+}
\AddCodeHighRule[latex/math]{1}{Brace}        {\[\{ \}] }
\AddCodeHighRule[latex/math]{7}{Bracket}      {\[\[ \] ]}
\AddCodeHighRule[latex/math]{3}{Parenthesis}{\[\( \) ]}
\AddCodeHighRule[latex/math]{9}{Comment}      {\%.*?\r}

\AddCodeHighRule[latex/table]{8}{Newline}     {\\\}
\AddCodeHighRule[latex/table]{1}{Alignment}{\&}
\AddCodeHighRule[latex/table]{6}{BeginEnd}     {\(begin|end)}
\AddCodeHighRule[latex/table]{4}{Command}      {\[A-Za-z]+}
\AddCodeHighRule[latex/table]{2}{Brace}        {\[\{ \}] }
\AddCodeHighRule[latex/table]{3}{Bracket}      {\[\[ \] ]}
\AddCodeHighRule[latex/table]{9}{Comment}      {\%.*?\r}

\AddCodeHighRule[latex/latex2]{1}{Argument}   {\#+\d}
\AddCodeHighRule[latex/latex2]{6}{NewCommand}{\(|e|g|x)def}
\AddCodeHighRule[latex/latex2]{5}{SetCommand}{\set[A-Za-z]+}
\AddCodeHighRule[latex/latex2]{4}{PrivateCmd}{\[A-Za-z@]*@[A-Za-z@]*}
\AddCodeHighRule[latex/latex2]{3}{Command}     {\[A-Za-z]+}
\AddCodeHighRule[latex/latex2]{2}{Brace}        {\[\{ \}] }
\AddCodeHighRule[latex/latex2]{7}{Bracket}      {\[\[ \] ]}
\AddCodeHighRule[latex/latex2]{9}{Comment}      {\%.*?\r}

\AddCodeHighRule[latex/latex3]{1}{Argument}   {\#+\d}
\AddCodeHighRule[latex/latex3]{2}{PrivateVar}{\[cgl]__[A-Za-z_:@]+}
\AddCodeHighRule[latex/latex3]{5}{PrivateFun}{\[_][A-Za-z_:@]+}
\AddCodeHighRule[latex/latex3]{4}{PublicVar}   {\[cgl]_[A-Za-z_:@]+}
\AddCodeHighRule[latex/latex3]{6}{PublicFun}   {\[A-Za-z_:@]+}
\AddCodeHighRule[latex/latex3]{8}{Brace}        {\[\{ \}] }
\AddCodeHighRule[latex/latex3]{3}{Bracket}      {\[\[ \] ]}
\AddCodeHighRule[latex/latex3]{9}{Comment}      {\%.*?\r}

```

2.7 Add CodeHigh Themes

```

\keys_define:nn {codehigh}
{
  theme .tl_set:N = \l__codehigh_theme_name_tl,
  theme .initial:n = default,
  style/main .code:n = \SetCodeHighStyle{main}{#1},
  style/code .code:n = \SetCodeHighStyle{code}{#1},
  style/demo .code:n = \SetCodeHighStyle{demo}{#1},
}

```

%% #1: theme name; #2: rule type; #3: styles

```

\NewDocumentCommand \SetCodeHighStyle {0{default} m m}
{
  \tl_set:cn {l__codehigh_style_#1_#2_tl} {#3}
}

\NewDocumentCommand \GetCodeHighStyle {0{default} m}
{
  \colorlet{codehigh@bg}{\tl_use:c {l__codehigh_style_#1_#2_tl}}
}

\SetCodeHighStyle[default]{main}{gray9}
\SetCodeHighStyle[default]{code}{gray9}
\SetCodeHighStyle[default]{demo}{white}

\SetCodeHighStyle[default]{0}{black}
\SetCodeHighStyle[default]{1}{brown3}
\SetCodeHighStyle[default]{2}{yellow3}
\SetCodeHighStyle[default]{3}{olive3}
\SetCodeHighStyle[default]{4}{teal3}
\SetCodeHighStyle[default]{5}{azure3}
\SetCodeHighStyle[default]{6}{blue3}
\SetCodeHighStyle[default]{7}{violet3}
\SetCodeHighStyle[default]{8}{purple3}
\SetCodeHighStyle[default]{9}{gray3}

```

2.8 Parse and Highlight Code

```

\int_new:N \l__codehigh_item_count_int
\tl_new:N \l__codehigh_code_to_parse_tl
\tl_new:N \l__codehigh_regex_match_type_tl
\tl_new:N \l__codehigh_regex_match_text_tl
\tl_new:N \l__codehigh_regex_before_text_tl

\cs_new_protected:Npn \__codehigh_parse_code:nN #1 #2
{
  \ifluatex
    \__codehigh_parse_code luatex:nN {#1} #2
  \else
    \__codehigh_parse_code normal:nN {#1} #2
  \fi
}

\cs_generate_variant:Nn \__codehigh_parse_code:nN {VN}

\cs_new_protected:Npn \__codehigh_parse_code_normal:nN #1 #2
{

```

```

\tl_set_eq:NN \l__codehigh_code_to_parse_tl #2
\bool_do_until:nn {\tl_if_empty_p:N \l__codehigh_code_to_parse_tl}
{
  \__codehigh_parse_code_once:nN {#1} \l__codehigh_code_to_parse_tl
  \int_compare:nNnTF {\l__codehigh_item_count_int} = {-1}
  {
    \__codehigh_typeset_text:nN {0} \l__codehigh_code_to_parse_tl
    \tl_clear:N \l__codehigh_code_to_parse_tl
  }
  {
    \tl_concat:NNN \l__codehigh_a_tl
      \l__codehigh_regex_before_text_tl \l__codehigh_regex_match_text_tl
    \str_remove_once:NV \l__codehigh_code_to_parse_tl \l__codehigh_a_tl
    %\tl_log:N \l__codehigh_code_to_parse_tl
    \__codehigh_typeset_text:nN {0}
      \l__codehigh_regex_before_text_tl
    \__codehigh_typeset_text:VN \l__codehigh_regex_match_type_tl
      \l__codehigh_regex_match_text_tl
  }
}
}

\cs_new_protected:Npn \__codehigh_parse_code_once:nN #1 #2
{
  \int_set:Nn \l__codehigh_item_count_int { -1 }
  \tl_clear:N \l__codehigh_regex_match_text_tl
  \tl_clear:N \l__codehigh_regex_before_text_tl
  \int_step_inline:nn {\cs:w \l__codehigh_#1_rule_count_int \cs_end:}
  {
    \regex_extract_once:cVNT {\l__codehigh_#1_#1_regex} #2 \l_tmpa_seq
    {
      \seq_get:NN \l_tmpa_seq \l__codehigh_m_tl
      \regex_split:cVNT { \l__codehigh_#1_#1_regex } #2 \l_tmpb_seq
      {
        \seq_get:NN \l_tmpb_seq \l__codehigh_b_tl
        \tl_set:Nx \l__codehigh_c_tl {\str_count:N \l__codehigh_b_tl}
        \bool_lazy_or:nnT
          { \int_compare_p:nNn {\l__codehigh_item_count_int} = {-1} }
          {
            \int_compare_p:nNn
              {\l__codehigh_item_count_int} > {\l__codehigh_c_tl}
          }
        {
          \int_set:Nn \l__codehigh_item_count_int {\l__codehigh_c_tl}
          \tl_set_eq:NN \l__codehigh_regex_before_text_tl
            \l__codehigh_b_tl
        }
      }
    }
  }
}

```



```

\l_set_eq:NN \l__codehigh_regex_match_text_tl
\l__codehigh_m_tl
\l_set_eq:Nc \l__codehigh_regex_match_type_tl
{l__codehigh_#1_#1_type_tl}
}
}
}
}
}

\ifluatex \directlua{require("codehigh.lua")} \fi

\cs_new_protected:Npn \__codehigh_parse_code luatex:nN #1 #2
{
\directlua{ParseCode(token.scan_argument(), token.scan_argument())}{#1}{#2}
%\tl_log:N \l__codehigh_parse_code_count_tl
\int_step_inline:nn {l__codehigh_parse_code_count_tl}
{
\__codehigh_typeset_text:vc
{l__codehigh_parse_style_#1_tl} {l__codehigh_parse_code_#1_tl}
}
}

%% #1: rule type, #2: text
\cs_new_protected:Npn \__codehigh_typeset_text:nN #1 #2
{
\group_begin:
\regex_replace_all:nnN { \r } { \c{par} \c{leavevmode} } #2
\ifluatex\else
\regex_replace_all:nnN { \ } { \c{relax} \c{space} } #2
\fi
\color{\tl_use:c {l__codehigh_style_ \l__codehigh_theme_name_tl _#1_tl}}
%\obeyspaces
#2
\group_end:
}

\cs_generate_variant:Nn \__codehigh_typeset_text:nN { VN, vc }

```

2.9 Don't Highlight Code

```

\cs_new_protected:Npn \__codehigh_parse_code_lite:N #1
{
\regex_replace_all:nnN { \r } { \c{par} \c{leavevmode} } #1
\regex_replace_all:nnN { \ } { \c{relax} \c{space} } #1
\tl_use:N #1
}

```