codeanatomy – Draw Code Anatomy

Usage

Hồng-Phúc Bùi †

Released 2023/01/24

Contents

1 Introduction 1

2 Tutorial 1
   2.1 Package Usage .................................................. 1
   2.2 Create an anatomy environment .................................. 2
   2.3 Typeset Code Listing in a TikZ-Node .......................... 2
   2.4 Mark Parts of Code ................................................. 4
   2.5 Create Annotation Labels ......................................... 4

3 Usage in conjunction with listings 6

4 Customize style 6

Change History 7

1 Introduction

The idea of this Package is to typeset illustrations of pieces of code with annotations on each single parts of code (Code Anatomy). The origin of this idea is code illustrations in the textbook [1]. This package just provides tool to draw those figures.

2 Tutorial

In this tutorial we will draw an anatomy of a function like the figure 1 step by step.

2.1 Package Usage

To use this package, just insert \usepackage{codeanatomy} in your \LaTeX{} file.

*This file describes v0.4-Beta, last revised 2023/01/24.
†E-mail: hong-phuc.bui (at) hwssar dot de
2.2 Create an anatomy environment

Next step is to create a `tikzpicture` environment. All commands in this package must be placed in a `tikzpicture` environment with option `remember picture`.

\begin{tikzpicture}
\begin{codeBlock}
function gcd(p, q) {
  if (q === 0) {
    return p;
  } else {
    return gcd(q, p % q);
  }
}
\end{codeBlock}
\end{tikzpicture}

The result of the above code is shown in the figure 2, which is not what we really want. All extra whitespaces and newlines in the listing are removed, further more { and } are interpreted as \LaTeX tokens and are not displayed.

We need to put `\ptab` and `\ ` into code to keep whitespaces and newlines. The characters { and } also need to be escaped by prefixing a `\` before them.

2.3 Typeset Code Listing in a TikZ-Node

As next step we need to put the piece of code in the `tikzpicture` environment using the command `\codeBlock`.

\begin{tikzpicture}
\begin{codeBlock}
function gcd(p, q) {
  if (q === 0) {
    return p;
  } else {
    return gcd(q, p % q);
  }
}
\end{codeBlock}
\end{tikzpicture}
The result (figure 3) is much more like what we expect than the version before (figure 2).

Function gcd:

```javascript
function gcd(p, q) {
    if (q === 0) {
        return p;
    } else {
        return gcd(q, p%q);
    }
}
```

Figure 3: Formatted Function
2.4 Mark Parts of Code

Now we can mark interesting parts of code with a blue boxes created by \cPart. At some positions we can use \[<\text{length}>\] to add a little amount of vertical space, so that the boxes do not touch each others.

\begin{tikzpicture}
\[\text{function}\gcd\{\text{(p, q)}\}\{\]
\[\text{if (q === 0) \{ \]
\[\text{return p; \]}
\[\text{\} else \{ \]
\[\text{return gcd(q, p\%q); \]}
\[\text{\} \]
\end{tikzpicture}

Figure 4: Function with marked parts

2.5 Create Annotation Labels

We can use \codeAnnotation to create annotation labels for each parts of code. To draw an arrow from label to a code part we can use the Ti\text{K}Z command \draw[\text{-},\text{annotation}] \{(\text{annotation label}) \rightarrow (\text{code part})\;.

Whereas \{(\text{annotation label})\}s are the first argument of \codeAnnotations and \{(\text{code part})\}s are the first argument of \cParts.
\begin{tikzpicture}[remember picture]
\node at (-0.5,-0.5) \[on background layer\]\draw[code grid debug] (-0.5,-0.5) grid (6.5,4.5);\}
\codeBlock{%\cPart{functionHead} {function \cPart{functionName}{gcd} \cPart{paramList}{(p, q)}} \{
\\[2.5pt]\ptab{}\mtPoint{mostLeft} if (q === 0) \{ \ \\
\ptab\ptab{} return p; \\\n\ptab\} else \{ \ \\
\ptab\ptab{} return gcd(q, p\%q); \extremPoint{mostRight} \ \\
\ptab\mbPoint{mostBottom}\} \} \}
\fitExtrem{functionBody}{(mostLeft) (mostRight) (mostBottom)}
\% Annotations
\codeAnnotation{functionHeadText}(-1,3){Function\head}
\codeAnnotation{functionBodyText}(-1,1){Function\body}
\codeAnnotation{functionNameText}( 1,4){Function\name}
\codeAnnotation{paramListText} ( 3,4){Parameter\list}
\% Annotation labels to code parts
\draw[-,annotation] (functionHeadText) -- (functionHead);
\draw[-,annotation] (functionBodyText) -- (functionBody);
\draw[-,annotation] (functionNameText) -- (functionName);
\draw[-,annotation] (paramListText) -- (paramList);
\end{tikzpicture}

Instead of operator -- we can use operator to \[\{(TikZ options)\}\] to draw a path from ((annotation label)) to ((code part)). Finally we can remove the command \draw[code grid debug]... at the begin of the tikzpicture. The final result is shown in the figure 5, which is almost the same as figure 1.

\begin{figure}[h]
\centering
\begin{tikzpicture}
\node at (0,0) {function \texttt{gcd} \{p, q\} \{\
if (q === 0) \{
return p;
\} else \{\
return gcd(q, p\%q);
\}
\}};
\end{tikzpicture}
\caption{Function with Annotation Labels}
\end{figure}
3 Usage in conjunction with listings

As we see in the previous section, the command \texttt{codeBlock} cannot typeset whitespaces correctly as we expect. A way to typeset code listing is using the package \texttt{listings}. See \texttt{codeanatomy.lstlisting.pdf} for more information.

4 Customize style

Maybe we want to highlight the function name \texttt{gcd} with some background color like figure 6. The best way to do this task is to assign this highlight format to a Ti\textit{k}Z style \texttt{jsid} and apply the style to the highlighted nodes. So they have the same format.

```latex
function \texttt{gcd}(p, q) \{
  if (q === 0) {
    return p;
  } else {
    return \texttt{gcd}(q, p \% q);
  }
\}
```

Function can call itself.

\begin{tikzpicture}[remember picture]
  \tikzstyle{\texttt{jsid}} = [code part, fill=red!10]
  \codeBlock{
    \cPart{functionHead}{function \texttt{gcd}(p, q) \{
      \ifnum q=0 \{
        return p;
      \} else {
        return \texttt{gcd}(q, p \% q);
      } \}
    }
  }
  \fitExtrem{functionBody}{(mostLeft) (mostRight) (mostBottom)}

  % Annotations
  \codeAnnotation{functionHeadText}{(-1,3)} {Function\Head}
  \codeAnnotation{functionBodyText}{(-1,1.5)} {Function\Body}
  \codeAnnotation{functionNameText}{(1,4)} {Function Name}
  \codeAnnotation{parameterListText}{(4,4)} {Parameter List}

  % Annotation labels to code parts
  \draw[->,annotation] (functionBodyText) -- (functionBody);
  \draw[->,annotation] (functionHeadText) -- (functionHead);
  \draw[->,annotation] (functionNameText) -- (functionName);
  \draw[->,annotation] (parameterListText) -- (paramList);

  \draw[->,annotation] (recursiveText) to[out=175,in=-15] (functionName);
  \draw[->,annotation] (recursiveText) to[out=180,in=45] (recursive.north east);
\end{tikzpicture}

Figure 6: Highlighted name of the function
Change History

v0.4-Beta
General: Add Option \langle style \rangle to cPart and iPart ............... 6
References