The vomit Class

Eito YONEYAMA

2025/09/02 v1.0

Abstract

This class replaces characters with the emoji 🤮. It is mainly a demonstration of hooking into LuaTeX callbacks.

Contents

1 Introduction

The vomit class intercepts input lines with Lua callbacks and replaces every non-command character with the emoji 🤮. It requires LuaLaTeX and the bxcoloremoji package.

2 Usage

Basic usage:

```
\documentclass{vomit}
\begin{document}
Hello, world!
\end{document}
```

To temporarily disable vomit mode, use `\stopvomit`. To re-enable it, use `\startvomit`. After calling `\startvomit`, all non-command characters are replaced by the vomit emoji 🤮.

3 Implementation

```
1 ⟨∗class⟩
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesClass{vomit}[2025/09/02 v1.0 Universal Emoji Replacer Class]
4
```

```
5 \LoadClass{article}
6 \RequirePackage{bxcoloremoji}
7 \RequirePackage{luacode}
8
9 \begin{luacode*}
10 local function utf8_len(s)
11   if type(s) ~= "string" then return 0 end
12   if unicode and unicode.utf8 and unicode.utf8.len then
13     local ok, v = pcall(unicode.utf8.len, s)
14     if ok and v then return v end
15   end
16   if utf8 and utf8.len then
17     local ok, v = pcall(utf8.len, s)
18     if ok and v then return v end
19   end
20   local n = 0
21   for _ in s:gmatch("[%z\1-\127\194-\244][\128-\191]*") do n = n + 1 end
22   return n
23 end
24
25 local function utf8_bytepos(s, n)
26   if n < 1 then return nil end
27   if unicode and unicode.utf8 and unicode.utf8.offset then
28     local ok, pos = pcall(unicode.utf8.offset, s, n)
29     if ok then return pos end
30   end
31   if utf8 and utf8.offset then
32     local ok, pos = pcall(utf8.offset, s, n)
33     if ok then return pos end
34   end
35   local i = 0
36   for pos, _ in s:gmatch("()([%z\1-\127\194-\244][\128-\191]*)") do
37     i = i + 1
38     if i == n then return pos end
39   end
40   return nil
41 end
42
43 local function utf8_sub_chars(s, i, j)
44   local len = utf8_len(s)
45   if i < 1 then i = 1 end
46   if j == nil or j > len then j = len end
47   if i > j then return "" end
48   local b1 = utf8_bytepos(s, i)
49   local b2next = utf8_bytepos(s, j+1)
50   local b2 = (b2next and (b2next - 1)) or #s
51   if not b1 then return "" end
52   return string.sub(s, b1, b2)
53 end
54
55 local function trim(s)
56   return (s:gsub("^%s+", ""):gsub("%s+$", ""))
57 end
58
```

```lua
59 local function vomitify_text(line)
60   if type(line) ~= "string" then
61     return line
62   end
63
64   if line:find("\\end{document}", 1, true) then
65     return line
66   end
67
68   local t = trim(line)
69   if t:match("^\\") or t == "" or t:match("^%%") then
70     return line
71   end
72
73   local len = utf8_len(line)
74   local i = 1
75   local res = {}
76
77   while i <= len do
78     local ch = utf8_sub_chars(line, i, i)
79
80     if ch == "\\" then
81       local cmd = ch
82       local j = i + 1
83       while j <= len do
84         local nch = utf8_sub_chars(line, j, j)
85         if nch:match("^[A-Za-z]$") then
86           cmd = cmd .. nch
87           j = j + 1
88         else
89           break
90         end
91       end
92       if cmd == "\\" and j <= len then
93         local next_ch = utf8_sub_chars(line, j, j)
94         if not next_ch:match("^[A-Za-z]$") then
95           cmd = cmd .. next_ch
96           j = j + 1
97         end
98       end
99       table.insert(res, cmd)
100      i = j
101    elseif ch == "{" or ch == "}" or ch == "&" or ch == "$" or ch:match("^%s$") then
102      table.insert(res, ch)
103      i = i + 1
104    else
105      table.insert(res, "\\coloremojicode{1F92E}")
106      i = i + 1
107    end
108  end
109
110  return table.concat(res)
111 end
112
```

```
113 function enable_vomit_mode()
114   if luatexbase and luatexbase.add_to_callback then
115     luatexbase.add_to_callback("process_input_buffer", vomitify_text, "vomitify_all_text")
116   else
117     callback.register("process_input_buffer", vomitify_text)
118   end
119 end
120
121 function disable_vomit_mode()
122   if luatexbase and luatexbase.remove_from_callback then
123     luatexbase.remove_from_callback("process_input_buffer", "vomitify_all_text")
124   else
125     callback.register("process_input_buffer", nil)
126   end
127 end
128 \end{luacode*}
129
130 \AtBeginDocument{%
131   \directlua{enable_vomit_mode()}%
132 }
133
134 \newcommand{\stopvomit}{%
135   \directlua{disable_vomit_mode()}%
136 }
137
138 \newcommand{\startvomit}{%
139   \directlua{enable_vomit_mode()}%
140 }
141 ⟨/class⟩
```