

# The `secnum` package

Gau, Syu

*Last Update: 2021/08/28*

## Abstract

The package `secnum` provides a marco `\setsecnum` which allows user to format section numbering intuitively.

## Contents

<b>A</b>	<b>Example</b>	<b>1</b>
<b>B</b>	<b>Usage</b>	<b>1</b>
	1 Set numbering format . . . . .	1
	2 Breaking the numbering . . . . .	2
	3 Package options . . . . .	2
<b>C</b>	<b>Process</b>	<b>3</b>
<b>D</b>	<b>Implementation</b>	<b>3</b>
	1 Preparations . . . . .	3
	2 Package option . . . . .	4
	3 Main function . . . . .	4
	4 Unabbrevation . . . . .	5
	5 Split to sequence . . . . .	5
	6 Read formatting info . . . . .	6
	7 Formatting . . . . .	7

## A Example

This document uses the following setting of section numbering format.

```
\usepackage[tocdep=2]{secnum}
\setsecnum{A,1.i}
```

## B Usage

Before using the macro, load the package in preamble.

```
\usepackage{secnum}
```

## 1 Set numbering format

One can format the section numbering by using the marco `\setsecnum` in preamble.

---

`\setsecnum` `\setsecnum{num format}`

A typical *num format* is like this:

A,1.i

It consists of some syntax abbrs of numbering formats, reffering the follows,

A	a	I	i	1
<code>\Alph</code>	<code>\alph</code>	<code>\Roman</code>	<code>\roman</code>	<code>\arabic</code>

and some separators delimiting them.

The separators can be any character except above abbrs, the tokens “{”, “}” and “#” (more precisely, explicit character tokens with category code 1 (begin-group) or 2 (end-group), and tokens with category code 6) and the space “`\` ”.

Note that *num format* must end with an abbr.

**T<sub>E</sub>Xhackers note:** This command will overwrite `secnumdepth` and `tocdepth`

## 2 Breaking the numbering

The comma “,” in above example is used as the breaking mark: for deep levels (in our case, deeper than sections), the numbering before “,” will be hidid.

Note that the breaking mark must immediately follows an abbr.

## 3 Package options

### 3.i tocdep

There is an option setting `tocdepth`, the table-of-contents depth manually.

---

`tocdep` `tocdep = integer`

The *integer* refers to the table-of-contents depth, which should between 1 and 5.

**T<sub>E</sub>Xhackers note:** If this option is used, then `\setsecnum` will not overwrite `tocdepth`.

### 3.ii breaking

Another option is used to change the breaking mark.

---

`breaking` `breaking = token`

The *token* will be the breaking mark (the default is the comma “,”). It can be any character except above abbrs, the tokens “{”, “}” and “#” (more precisely, explicit character tokens with category code 1 (begin-group) or 2 (end-group), and tokens with category code 6) and the space “`\` ”.

## C Process

The process of the macro `\setsecnum` can be explained as follows.

- Step 1. The main function eats the input, saying `A,1.i`, and stores it in a token list.
- Step 2. Replace abbrs by macros. In our example, it results “`\Alph,\arabic.\roman`”
- Step 3. Split this token list into a sequence by macros. In our example, it results “`\Alph`”, “`,\arabic`” and “`.\roman`”.
- Step 4. Store those codes in individual containers.
- Step 5. Detect if there is `\thechapter`. Skip the chapter level if not. In our example, this is the case.
- Step 6. Use the containers to redefine `\thesection`, `\thesubsection`, `\thesubsubsection` etc. In each step, detect if such level needs numbering and if there is a breaking mark in the container. In our example, the numbering formats will be redefined as

```
\renewcommand*{\thesection}{\Alph{section}}
\renewcommand*{\thesubsection}{\arabic{subsection}}
\renewcommand*{\thesubsubsection}{thesubsection.\roman{subsubsection}}
```

## D Implementation

The following is the implementation. Users can ignore.

### 1 Preparations

This package uses L<sup>A</sup>T<sub>E</sub>X3. Therefore, the packages `expl3`, `xparse` and `l3keys2e` are needed and should use `\ProvidesExplPackage` rather than `\ProvidesPackage`.

```
1 <*package>
2 <@@=syu>
3 \NeedsTeXFormat{LaTeX2e}
4 \RequirePackage{expl3}
5 \ProvidesExplPackage{secnum}{2021/08/28}{ }
6   { An intuitive way to format section numbering }
7 \RequirePackage{xparse,l3keys2e}
```

`\l__syu_secnum_tl` The variables are used to store the formatting information.

```
\l__syu_secnum_seq 8 \tl_new:N \l__syu_secnum_tl
9 \seq_new:N \l__syu_secnum_seq
10 \int_new:N \l__syu_secnum_depth
```

`\g__syu_chapter_tl` The following variables are used to store the individual formatting codes.

```
\g__syu_section_tl 11 \tl_new:N \g__syu_chapter_tl
\g__syu_subsection_tl 12 \tl_new:N \g__syu_section_tl
\g__syu_subsubsection_tl 13 \tl_new:N \g__syu_subsection_tl
\g__syu_paragraph_tl 14 \tl_new:N \g__syu_subsubsection_tl
\g__syu_subparagraph_tl 15 \tl_new:N \g__syu_paragraph_tl
16 \tl_new:N \g__syu_subparagraph_tl
```

`\g__syu_if_thechapter_int` This *<integer>* encodes if `\thechapter` is defined.

```
17 \int_new:N \g__syu_if_thechapter_int
If \thechapter is defined, it is 1.
18 \if_cs_exist:N \thechapter
19   \int_gset:Nn \g__syu_if_thechapter_int 1
Otherwise, it is 0.
20 \else:
21   \int_gset:Nn \g__syu_if_thechapter_int 0
22 \fi:
```

`\l__syu_secnum_bkm` This variable is used to store the breaking mark.

```
23 \tl_new:N \g__syu_secnum_bkmr
24 \tl_gset:Nx \g__syu_secnum_bkmr {,}
Note that one needs the following variants
25 \cs_generate_variant:Nn \tl_if_in:NnTF { NV }
26 \cs_generate_variant:Nn \tl_remove_all:Nn { NV }
```

## 2 Package option

```
27 \keys_define:nn { syu / options }
28   {
```

**tocdep** Set the table-of-contents depth.

```
29   tocdep .code:n =
30   {
31     \int_const:Nn \g__syu_tocdep {#1}
32     \setcounter{tocdepth}{ \g__syu_tocdep }
33   },
```

**breaking** Set the breaking mark used in *<num format>*.

```
34   breaking .code:n =
35   {
36     \tl_gset:Nx \g__syu_secnum_bkmr {#1}
37   },
38 }
```

Passing keys to options.

```
39 \ProcessKeysOptions{ syu / options }
```

## 3 Main function

**\setsecnum** Here is the definition of the main function `\setsecnum`.

```
40 \DeclareDocumentCommand{\setsecnum}{m}
41   {
```

Store the input in.

```
42   \tl_set:Nn \l__syu_secnum_tl {#1}
```

Replace syntax abbrs by corresponding macros.

```
43   \__syu_secnum_unabbr:N \l__syu_secnum_tl
```

Split into a sequence by macros.

```
44 \__syu_split_by_macros:NN \l__syu_secnum_tl \l__syu_secnum_seq
```

Read formatting information.

```
45 \__syu_secnum_from_seq:N \l__syu_secnum_seq
```

Set the `secnumdepth` and `tocdepth`.

```
46 \int_set:Nn \l__syu_secnum_depth
47 {
48   \seq_count:N \l__syu_secnum_seq
49 }
50 \setcounter{secnumdepth}
51 {
52   \int_eval:n
53   {
54     \l__syu_secnum_depth - \g__syu_if_thechapter_int
55   }
56 }
57 \int_if_exist:NTF \g__syu_tocdep
58 {
59   \setcounter{tocdepth}{ \g__syu_tocdep }
60 }
61 {
62   \setcounter{tocdepth}
63   {
64     \int_eval:n
65     {
66       \l__syu_secnum_depth - \g__syu_if_thechapter_int
67     }
68   }
69 }
```

Format numberings.

```
70 \__syu_secnum:
71 }
```

## 4 Unabbravation

`\__syu_secnum_unabbr:N` This function replace the abbrs in a  $\langle tl var \rangle$  by expansions.

```
72 \cs_new_protected:Npn \__syu_secnum_unabbr:N #1
73 {
74   \regex_replace_all:nnN {A} {\c{Alpha}} #1
75   \regex_replace_all:nnN {a} {\c{alpha}} #1
76   \regex_replace_all:nnN {I} {\c{Roman}} #1
77   \regex_replace_all:nnN {i} {\c{roman}} #1
78   \regex_replace_all:nnN {1} {\c{arabic}} #1
79 }
```

## 5 Split to sequence

`\__syu_split_by_macros:NN` This function split a  $\langle tl var \rangle$  into a  $\langle sequence \rangle$  by macros.

```
80 \cs_new_protected:Npn \__syu_split_by_macros:NN #1 #2
81 {
82   \tl_clear:N \l__tmpa_tl
```

```

83 \seq_clear:N #2
84 \tl_map_inline:Nn #1
85   {
86     \tl_put_right:Nn \l_tmpa_tl ##1
87     \__syu_if_macro:nT ##1
88     {
89       \seq_put_right:NV #2 \l_tmpa_tl
90       \tl_clear:N \l_tmpa_tl
91     }
92   }
93 }

```

But how to see if an *<item>* in the token list is a macro?

`\g__syu_macro_tl` This *<tl var>* stores the first five characters of the meaning of any macro, i.e. `macro` (watch out its catcode). The idea is to create a *<tl var>* and then set its value to be the first five characters of its meaning.

```

94 \tl_new:N \g__syu_macro_tl
95 \tl_set:Nx \g__syu_macro_tl { \meaning \g__syu_macro_tl }
96 \tl_gset:Nx \g__syu_macro_tl { \tl_range:Nnn \g__syu_macro_tl {1}{5} }

```

`\__syu_if_macro:nT` Then, define a conditional testing if the input is a macro. Note that I use `\if_meaning` rather than `\tl_if_eq:NNTF`.

```

\__syu_if_macro:nF
\__syu_if_macro:nTF
97 \prg_new_protected_conditional:Npnn \__syu_if_macro:n #1 { T , F , TF }
98   {
99     \group_begin:
100     \tl_set:Nx \l_tmpa_tl {\meaning #1}
101     \tl_set:Nx \l_tmpa_tl {\tl_range:Nnn \l_tmpa_tl {1} {5}}

```

This is a trick to keep `\l_tmpa_tl` in the current local group

```

102 \exp_after:wN
103 \group_end:

```

while throwing the comparison result out.

```

104 \if_meaning:w \l_tmpa_tl \g__syu_macro_tl
105 \prg_return_true:
106 \else:
107 \prg_return_false:
108 \fi:
109 }

```

## 6 Read formatting info

`\__syu_secnum_from_seq:N` Read the formatting info from given *<sequence>*.

```

110 \cs_new_protected:Npn \__syu_secnum_from_seq:N #1
111   {

```

Use `\tl_gset:Nx` since: 1, these data are global and 2: I need them eating the fully expanded results.

```

112 \tl_gset:Nx \g__syu_chapter_tl
113   { \seq_item:Nn #1 { \g__syu_if_thechapter_int } }
114 \tl_gset:Nx \g__syu_section_tl
115   { \seq_item:Nn #1 { 1 + \g__syu_if_thechapter_int } }
116 \tl_gset:Nx \g__syu_subsection_tl

```

```

117     { \seq_item:Nn #1 { 2 + \g__syu_if_thechapter_int } }
118   \tl_gset:Nx \g__syu_subsubsection_tl
119     { \seq_item:Nn #1 { 3 + \g__syu_if_thechapter_int } }
120   \tl_gset:Nx \g__syu_paragraph_tl
121     { \seq_item:Nn #1 { 4 + \g__syu_if_thechapter_int } }
122   \tl_gset:Nx \g__syu_subparagraph_tl
123     { \seq_item:Nn #1 { 5 + \g__syu_if_thechapter_int } }
124 }

```

## 7 Formatting

\\_\_syu\_secnum: Formatting section numbering.

```

125 \cs_new:Nn \__syu_secnum:
126 {

```

### 7.i Detect if there is \thechapter

When \thechapter is defined, start from it.

```

127   \if_cs_exist:N \thechapter
128     \renewcommand*{\thechapter}
129     { \g__syu_chapter_tl {chapter} }

```

Test if the numbering breaks before section.

```

130   \tl_if_in:NVTF \g__syu_section_tl \g__syu_secnum_bkmr
131   {
132     \tl_remove_all:NV \g__syu_section_tl \g__syu_secnum_bkmr
133     \renewcommand*{\thesection}
134     { \g__syu_section_tl {section} }
135   }
136   {
137     \renewcommand*{\thesection}
138     {
139       \thechapter
140       \g__syu_section_tl {section}
141     }
142   }

```

Otherwise start from \thesection.

```

143   \else:
144     \renewcommand*{\thesection}
145     { \g__syu_section_tl {section} }
146   \fi:

```

### 7.ii Subsections

Test if the subsections are needed to be numbered.

```

147   \tl_if_empty:NTF \g__syu_subsubsection_tl
148   {}
149   {

```

Test if the numbering breaks before subsection.

```

150     \tl_if_in:NVTF \g__syu_subsubsection_tl \g__syu_secnum_bkmr
151     {
152       \tl_remove_all:NV \g__syu_subsubsection_tl \g__syu_secnum_bkmr

```

```

153         \renewcommand*{\thesubsection}
154         { \g__syu_subsection_tl {subsection} }
155     }
156     {
157         \renewcommand*{\thesubsection}
158         {
159             \thesection
160             \g__syu_subsection_tl {subsection}
161         }
162     }
163 }

```

### 7.iii Subsubsections

Test if the subsubsections are needed to be numbered.

```

164     \tl_if_empty:NTF \g__syu_subsubsection_tl
165     {}
166     {

```

Test if the numbering breaks before subsubsection.

```

167         \tl_if_in:NVTF \g__syu_subsubsection_tl \g__syu_secnum_bkmr
168         {
169             \tl_remove_all:NV \g__syu_subsubsection_tl \g__syu_secnum_bkmr
170             \renewcommand*{\thesubsubsection}
171             { \g__syu_subsubsection_tl {subsubsection} }
172         }
173         {
174             \renewcommand*{\thesubsubsection}
175             {
176                 \thesubsection
177                 \g__syu_subsubsection_tl {subsubsection}
178             }
179         }
180     }

```

### 7.iv Paragraphs

Test if the paragraphs are needed to be numbered.

```

181     \tl_if_empty:NTF \g__syu_paragraph_tl
182     {}
183     {

```

Test if the numbering breaks before paragraph.

```

184         \tl_if_in:NVTF \g__syu_paragraph_tl \g__syu_secnum_bkmr
185         {
186             \tl_remove_all:NV \g__syu_paragraph_tl \g__syu_secnum_bkmr
187             \renewcommand*{\theparagraph}
188             { \g__syu_paragraph_tl {paragraph} }
189         }
190         {
191             \renewcommand*{\theparagraph}
192             {
193                 \thesubsubsection
194                 \g__syu_paragraph_tl {paragraph}

```



```

195         }
196     }
197 }

```

## 7.v Subparagraphs

Test if the subparagraphs are needed to be numbered.

```

198 \tl_if_empty:NTF \g__syu_subparagraph_tl
199 {}
200 {

```

Test if the numbering breaks before paragraph.

```

201     \tl_if_in:NVTF \g__syu_subparagraph_tl \g__syu_secnum_bkmr
202     {
203         \tl_remove_all:NV \g__syu_subparagraph_tl \g__syu_secnum_bkmr
204         \renewcommand*{\thesubparagraph}
205         { \g__syu_subparagraph_tl {subparagraph} }
206     }
207     {
208         \renewcommand*{\thesubparagraph}
209         {
210             \theparagraph
211             \g__syu_subparagraph_tl {subparagraph}
212         }
213     }
214 }
215 }

```

```

216 \end{package}

```