Grundgesetze.sty for LaTeX2e Documentation

Marcus Rossberg
University of Connecticut
marcus.rossberg@uconn.edu

Version 1.01 2014/03/22

Grundgesetze.sty is a LaTeX2e package for typesetting Gottlob Frege's *begriffsschrift* [concept-script] formalism in the style of his *Grundgesetze der Arithmetik* (1893/1903). *Grundgesetze.sty* was developed for the 2013 English edition of Frege's book.[1] The package is based on Josh Parsons's *begriff.sty* which renders the formalism in the style of Frege's earlier work, *Begriffsschrift* (1879). It was amended by Richard G. Heck Jr., J. J. Green, Agustín Rayo, and Marcus Rossberg. Thanks to Philip A. Ebert for testing, comments, and suggestions. Note that Frege's defined function symbols are not rendered by this package, but rather by J. J. Green's *fge.sty*.

1 Options

At present the only package option is `bguq`, which causes the package to use the `bguq` font for an alternative universal quantifier (concavity). Of course, one must have the `bguq` font installed to use this option, but it is included in recent versions of the big TeX distrubtions.

2 Basic Commands

`\GGhorizontal` The horizontal, —

`\GGnot` The negation-stroke, $\vdash$

`\GGconditional` Conditional-stroke: called as `\GGconditional{p}{q}` yields $\begin{smallmatrix}\vdash q\\ \llcorner p\end{smallmatrix}$ (i.e., '$p \supset q$')

`\GGquant` Concavity: called as `\GGquant{\mathfrak a}` gives $\smile\!\!\!\!\!\!\frown^{\mathfrak{a}}$ (i.e., universal quantifier, '$\mathfrak{a}$' is the quantified variable)

`\GGjudge` Judgement-stroke, $\vdash$

`\GGdef` Definition-stroke, $\Vdash$

`\GGbracket` Automatically scaling brackets, `\GGbracket{\ldots}` yields $(\ldots)$ (see examples)

`\GGsqbracket` Analogous square brackets, $[\ldots]$

A complete list of commands and compatibility synonymns in the package can be found in Table 4, and the lengths parameterising the appearance of the output in Table 5.

[1] Gottlob Frege: *Basic Laws of Arithmetic*. Translated and edited by Philip A. Ebert and Marcus Rossberg. Oxford 2013.

2.1 Examples

- `\GGjudge \GGquant{\mathfrak a} \mathfrak a = \mathfrak a`

  yields

$$\vdash \overset{\mathfrak{a}}{\smile} \mathfrak{a} = \mathfrak{a}$$

- `\GGjudge \GGnot \GGquant{\mathfrak F} \GGnot \GGquant{\mathfrak a} \mathfrak{Fa}`

  yields

$$\vdash \overset{\mathfrak{F}}{\smile} \overset{\mathfrak{a}}{\smile} \mathfrak{Fa}$$

- `\GGjudge \GGconditional{(\GGhorizontal p)}{p}`

  yields

$$\vdash \begin{array}{l} p \\ (\!\!- p) \end{array}$$

- `\GGjudge \GGbracket{\GGconditional{p}{q}} = \GGbracket{\GGconditional{\GGnot q}{\GGnot p}}`

  yields

$$\vdash \left(\begin{array}{l} q \\ p \end{array} \right) = \left(\begin{array}{l} p \\ q \end{array} \right)$$

There are further examples, including Frege's basic laws of logic, available for download on `www.frege.info`.

3 Advanced Typesetting

3.1 Left-alignment of terminal forumlae: `\GGterm`

Conditional-strokes, negation-strokes, and concavities that are embedded in conditionals can result in a ragged appearance of the formula:

- `\GGjudge\GGconditional{p}{\GGconditional{q}{p}}`

  yields:

$$\vdash \begin{array}{l} p \\ q \\ p \end{array}$$

- `\GGjudge\GGconditional{Fa} {\GGnot \GGquant{\mathfrak a} \GGnot F \mathfrak a}`

  yields:

$$\vdash \overset{\mathfrak{a}}{\smile} Fa \mathfrak{a} \\ Fa$$

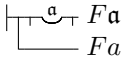
In Frege's original work, the component formulae of conditionals are left-aligned. This can be achieved by marking "terminal formulae" using the command `\GGterm{`⟨math⟩`}`; the length `\GGlinewidth` specifies the distance of the terminal formula from the left end of the whole formula (typically, '⊢'):

- ```
  \setlength{\GGlinewidth}{9.2pt} \GGjudge
  \GGconditional
       {\GGterm{p}}
       {\GGconditional{\GGterm{q}}
                      {\GGterm{p}}}
  ```

  yields:


- ```
  \setlength{\GGlinewidth}{25.2pt}
  \GGjudge\GGconditional{\GGterm{Fa}}
                        {\GGnot \GGquant{\mathfrak a} \GGnot
                         \GGterm{F \mathfrak a}}
  ```

  yields:



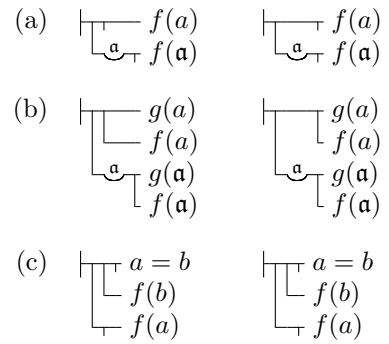
negation-stroke	⊤	4.4pt
conditional-stroke	⊤	4.4pt
concavity	◡	11.6pt
judgement-stroke:	⊢	
present		add .4pt
not present		subtract 2pt

Table 1: Lengths of embedded symbols

The correct values for `\GGlinewidth` for each formula can be determined by adding up the lengths of the embedded symbols, as given in Table 1, or by using a GUI that allows producing LaTeX and XML code for *begriffsschrift* formulae via mouse-click and that will calculate and output the correct values. The GUI is available for download on `www.frege.info`.

3.2 Adding horizontal lengths manually: `\GGnonot`, etc.

Readability is sometimes aided by moving, e.g., negations to the right end of the horizontal in a complex formula. For instance, Frege nearly always preferred the rendering displayed on the right in these types of formulae:



The right-hand formulae are produced by inserting commands for horizontals of the appropriate length directly at the position where the "space" should appear. The three right-hand formulae above are created in this way:

(a)
```
\GGjudge \GGconditional
         {\GGquant{\mathfrak a} \GGnot f(\mathfrak a)}
         {\GGnoquant \GGnot f(a)}
```

(b)
```
\GGjudge \GGconditional
         {\GGquant{\mathfrak a}
               \GGconditional{f(\mathfrak a)}{g(\mathfrak a)}}
         {\GGnoquant \GGconditional{f(a)}{g(a)}}
```

(c)
```
\GGjudge \GGconditional
         {\GGnonot \GGnot f(a)}
         {\GGconditional{\GGnonot f(b)}{\GGnot a=b}}
```

4 Comparison and compatibility with *begriff.sty*

Josh Parsons's *begriff.sty*, on which *grundgesetze.sty* is based, is closer in appearance to Frege's formalism as it is presented in Frege's first book, *Begriffsschrift* (1879). The corresponing commands were given different names so that both packages can be used in the same TEX document; see Table 2.

begriff.sty command	symbol	*grundgesetze.sty* symbol	command
`\BGcontent`	$-$	$—$	`\GGhorizontal`
`\BGnot`	$\vdash$	$\vdash$	`\GGnot`
`\BGconditional{p}{q}`	$\vdash\!\!\begin{array}{l}q\\p\end{array}$	$\vdash\!\!\begin{array}{l}q\\p\end{array}$	`\GGconditional{p}{q}`
`\BGquant{\mathfrak a}`	$\smile_{\mathfrak{a}}$	$\smile_{\mathfrak{a}}$	`\GGquant{\mathfrak a}`
`\BGassert`	$\vdash$	$\vdash$	`\GGjudge`
`\BGbracket{\ldots}`	$\left(\vdash\!\!\begin{array}{l}q\\p\end{array}\right)$	$\left(\vdash\!\!\begin{array}{l}q\\p\end{array}\right)$	`\GGbracket{\ldots}`

Table 2: Compatibility with *begriff.sty*

Also note the differences in alignment between `\BGbracket` and `\GGbracket` as shown in Table 3

`\BGbracket` $\vdash (\dot{\varepsilon} f(\varepsilon) = \dot{\alpha} g(\alpha)) = \smile_{\mathfrak{a}} \left(\begin{array}{l}\vdash\!\!\!— f(\mathfrak{a}) = g(\mathfrak{a})\\ \vdash\!\!\!\vdash \mathfrak{a} = \dot{\varepsilon} f(\varepsilon)\\ \vdash\!\!\!\vdash \mathfrak{a} = \dot{\alpha} g(\alpha)\end{array}\right)$

`\GGbracket`: $\vdash (\dot{\varepsilon} f(\varepsilon) = \dot{\alpha} g(\alpha)) = \smile_{\mathfrak{a}} \left(\begin{array}{l}\vdash\!\!\!— f(\mathfrak{a}) = g(\mathfrak{a})\\ \vdash\!\!\!\vdash \mathfrak{a} = \dot{\varepsilon} f(\varepsilon)\\ \vdash\!\!\!\vdash \mathfrak{a} = \dot{\alpha} g(\alpha)\end{array}\right)$

Table 3: `\BGbracket` and `\GGbracket` alignment

4.1 Conversion of a *begriff.sty* document into a *grundgesetze.sty* document

A straightforward way to convert the a LaTeX document that uses *begriff.sty* into one that uses *grundgesetze.sty* without manually exchanging the commands is to find and replace (using wrap search) "`\BG`" by "`\GG`". Synonyms have been added to *grundgesetze.sty* to allow the use of all *begriff.sty* commands "translated" in this way (see Table 4).

command	symbol	synonym / comment
`\GGterm{\ldots}`		(marks terminal formula)
`\GGhorizontal`	—	`\GGcontent`
`\GGjudge`	⊢	`\GGassert`
`\GGjudgelong`	⊢—	`\GGjudgealone`, `\GGassertlong`, `\GGassertalone`
`\GGjudgevar{`⟨*length*⟩`}`	⊢	`\GGassertvar{`⟨*length*⟩`}` (variable horizontal length, here: 6pt)
`\GGdef`	‖⊢	
`\GGdeflong`	‖⊢—	`\GGdefalone`
`\GGdefvar{`⟨*length*⟩`}`	‖⊢	(variable horizontal length, here: 6pt)
`\GGnot`	⊤	`\GGneg`
`\GGnotalone`	⊤	(standalone negation-stroke)
`\GGdnot`	⊤⊤	(standalone double negation-stroke)
`\GGconditional{p}{q}`	$\begin{smallmatrix}\top q\\ \lfloor p\end{smallmatrix}$	
`\GGquant{\mathfrak a}`	$\smile^{\mathfrak a}$	
`\GGall{a}`	$\smile^{\mathfrak a}$	`\GGquant{\mathfrak a}`
`\GGbracket{\ldots}`	(…)	(automatically scaling brackets)
`\GGsqbracket{\ldots}`	[…]	(ditto square brackets)
`\GGnonot`	–	horizontal of `\GGnot` length
`\GGnoquant`	—	horizontal of `\GGquant` length
`\GGnoboth`	——	horizontal of length: `\GGnot` plus `\GGquant`
`\GGnonotalone`	—	horizontal of `\GGnotalone` length
`\GGnodnot`	—	horizontal of `\GGdnot` length
`\GGoddspace`	—	horizontal of length: `\GGquant` minus `\GGnot`
`\GGtinyspace`	-	horizontal of length: `\GGquant` minus twice `\GGnot`)
`\GGtiniestspace`	-	horizontal of length: thrice `\GGnot` minus `\GGquant`

Table 4: All commands (and synonyms) defined by the package

length	default value	explanation
`\GGthickness`	0.4pt	thickness of horizontal and vertical lines
`\GGquantthickness`	0.75× `\GGthickness`	thickness of the line of the quantifier "dish"
`\beforelen`	2.4pt	length of horizontal before quantifier, conditional, and negation
`\GGafterlen`	2pt	length of horizontal after quantifier, conditional, negation, judgement-, and definition-stroke
`\GGspace`	3pt	space between right end of horizontal and terminal formula
`\GGlift`	2pt	lift of horizontal from baseline
`\GGlinewidth`	(n/a)	total length from left end of formula (typically, '`\GGjudge`') and the beginning of the terminal formula (see §3.1)

Table 5: Length parameters and their default values