The Cochineal Font Package

Michael Sharpe

November 19, 2023

Cochineal is a fork of Crimson, a remarkable creation of Sebastian Kosch inspired by oldstyle font designers. The name Cochineal is intended to suggest that, while it is crimson, there may be bugs involved. More than 1500 glyphs were added to Crimson, allowing a more than modest chance that some poor spacing or kerning or accent placement might have been introduced. Such problems will occur in the less frequented parts of the fonts, and I would appreciate reports of problems you may discover. You are unlikely to trip over issues if you stay within the bounds of, say, the T1 encoding. I did correct problems in a number of glyphs from Crimson that FontForge uncovered and that could have led to poor rendition on some platforms, especially MAC OS. I also changed the emsize to `1000` (the standard for PostScript-flavored Opentype) from the Crimson value of `1024` and rescaled accordingly.

Cochineal provides `Roman`, `Bold`, `Italic` and `BoldItalic`. (Crimson provided semibold weights but with only limited coverage, which I did not wish to extend as this would have required the creation of about 2000 additional glyphs. Bob Tennent's `Crimson` package does support semibold and the other styles, though without my glyph additions.)

Package Features:
In addition to the encodings `OT1`, `T1`, `TS1`, `LY1` in general use by Western European (and some Eastern European) scripts, the package also offers `LGR` with full support for monotonic, polytonic and some ancient Greek, and `T2A` and `OT2` Cyrillic support. All allow a choice from four figure styles—`TLF` (tabular lining figures, monospaced and uppercase), `LF` (proportional lining figures, uppercase), `TOsF` (tabular oldstyle figures, monospaced and lowercase) and `OsF` (proportional oldstyle figures, lowercase). The encodings `OT1`, `T1`, `TS1`, `LY1` offer Small Caps, even *Italics Small Caps*, and additional figure styles—superiors, inferiors and denominators. These features are available from either `fontspec` or from [pdf]LaTeX. In LaTeX, you access these through the macros `\textsu`, `\textinf` and `\textde`, or through their font-switching equivalents `\sufigures`, `\infigures` and `\defigures`. For example:

- `M\textsu{lle} Dupont` and `M{\sufigures lle} Dupont` both produce M$^{\text{lle}}$ Dupont.
- `{\infigures 12345}` and `\textinf{12345}` render as $_{12345}$, dipping noticeably below the baseline, while `{\defigures 12345}` and `\textde{12345}` render as 12345, aligned with the baseline.

As of version 1.062 (2020), there is a new `\textfrac` macro that works as in the following examples:

- (two arguments) `\textfrac{31}{32}` renders as $^{31}\!/_{32}$;
- (three arguments) `\textfrac[2]{63}{64}` renders as $2^{63}\!/_{64}$.

In addition, there is a `\textcircled` macro that makes `\textcircled{W}` render as Ⓦ. Numerals can also be circled: `\textcircled{2}` renders as ②.

If you load the `cochineal` package by means of option `cochineal` to the `newtx` package, version 1.71 or higher,

there is a stacked fraction construction available using the macro `\textsfrac` which behaves like `\textfrac` except with the fractional part stacked vertically rather than diagonally. For example, `\textsfrac[2]{5}{16}` produces $2\frac{5}{16}$.

Package Options and Macros:
The package defines two macros, `\useosf` and `\useproportional`, useable only in the preamble, which determine the default figure style in text. A typical invocation would be something like

```
\usepackage{cochineal} % default figure style is tabular, lining
% load sans and typewriter fonts
% load a math font---it will use tabular lining figures in math
\useosf % switch from lining figures to oldstyle figures
\useproportional % switch from tabular to proportional
```

There is a simpler way to achieve essentially the same result, but with the advantage that the figure styles are not loaded until after the math package (if any) is loaded, so that math always uses the default tabular lining figures.

```
% If you use babel, load it here, before cochineal
\usepackage[p,osf]{cochineal} % default figure style is proportional, oldstyle
% load sans and typewriter fonts
% load a math font---it will use tabular lining figures in math
```

As of version 1.71 of `newtx`, it may be more convenient to use its mechanism for loading `cochineal` as text and `newtxmath` as math.

```
% If you use babel, load it here, before newtx
% load sans and typewriter fonts
\usepackage[p,osf,cochineal]{newtx} % text figure style is proportional, oldstyle, math will use ta
% if you use unicode latex with
```

No matter what the default figure style in text, the package provides switches and macros to use any available figure style.

- `\textlf{}` and `{\lfstyle }` give proportional lining figures; `\texttlf{}` and `{\tlfstyle }` give tabular lining figures; `\textosf{}` and `{\osfstyle }` give proportional oldstyle figures; `\texttosf{}` and `{\tosfstyle }` give tabular oldstyle figures; `\textfrac{3}{4}` uses superior and denominator figures to make the fraction ¾.

The options that can be passed to `cochineal.sty` are the following:

- `scale` or `scaled`: a magnification factor—e.g., `scaled=1.02` enlarges all text controlled by the package by 2%;
- `p`, or `proportional`: make proportional figures the default rather than tabular; `lf`, or `lining`: make lining figures the default (this is already the default); `osf`, or `oldstyle`: make oldstyle figures the default rather than lining;
- `sups`: use superior figures to make footnote markers, rather than the LaTeX's default markers;
- `swashQ`: use Cochineal's swash Q instead of its tamer default version, Q;
- `scosf`: always use oldstyle figures within a small caps block;

- `theoremfont`: for theorem statements in the `plain` style, use a doctored version of italics that has upright figures, braces, brackets, parentheses, exclamation mark, colon and semicolon. It is implemented as the `slanted` shape, and there are two options you can add to the call to the `cochineal` package to modify the way it outputs figures.

  The default (neither of the following options is set) is to use upright figures within theorem text, but in the same alignment (proportional or tabular) and the same style (lining or oldstyle) as in general text.

  Option `thmtabular` causes figures in theorem text to render in tabular alignment, while option `thmlining` causes figure styles to render in lining rather than oldstyle. The two may be used in conjunction, forcing figures in theorem text to render as tabular lining figures.

Usage with unicode LaTeX

The cochineal package works with all latex engines, and has some extra tricks available only in unicode LaTeX. In addition, `newtx` also works with all latex engines and may be used to make a simplified call to use cochineal for text and `newtxmath` (with cochineal letters) for math. Two crucial options—`type1text` and `nofontspec`—each of which may be used with both `newtx` and `cochineal`, determine how unicode latex will process your `tex` document.

- Neither `type1text` (nor the equivalent `type1`) nor `nofontspec` specified: load `fontspec` and use `Cochineal` opentype fonts.
- `type1text` specified but not `nofontspec`: load `fontspec` but process text as `pdflatex` would. (Other subsidiary opentype fonts could be loaded using `fontspec` arguments.)
- `nofontspec` specified: `fontspec` not loaded, text processed as under `pdflatex`.

There are some more decorative glyph shapes as alternatives for Q and J in version 1.077 that are available only using unicode latex via `StylisticSet=3` and `StylisticSet=2` respectively. (The latter is available only in italic shapes.) They may be turned on as global options to `newtx` or `cochineal` by `altQ`, `altJ`.

```
\def\decorativeQ{{\addfontfeature{RawFeature=+ss03}Q}}
\def\decorativeJ{{\addfontfeature{RawFeature=+ss02}J}}
```

Q: `\decorativeQ`: Q
J: `\decorativeJ`: J
There is also an option `oldSS` you may specify to use the older form of `\SS` rather than the newer German Sharp S glyph.

Mathematical accompaniment

The package contains fonts for use as math letters that are derived from Cochineal Roman and Greek glyphs and the `newtxmath` family. Note that υ and ν (Greek `nu`) are quite distinct. Here's a sample.

```
% preamble should include, in this order:
\usepackage[T1]{fontenc}
% load babel here
\usepackage[p,osf]{cochineal}
\usepackage[varqu,varl,var0]{inconsolata}
\usepackage[scale=.95,type1]{cabin}
```

```
\usepackage[cochineal,vvarbb]{newtxmath}
\usepackage[cal=boondoxo]{mathalfa}
```

The typeset math below follows the ISO recommendations that only variables be set in italic. Note the use of upright shapes for d, e and π. (The first two are entered as `\mathrm{d}` and `\mathrm{e}`, and in fonts derived from `mtpro2` or `newtxmath`, the latter is entered as `\uppi`.)

Simplest form of the *Central Limit Theorem*: *Let $X_1, X_2, \cdots$ be a sequence of iid random variables with mean* 0 *and variance* 1 *on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Then*

$$\mathbb{P}\left(\frac{X_1 + \cdots + X_n}{\sqrt{n}} \le v\right) \to \mathfrak{N}(v) := \int_{-\infty}^{v} \frac{\mathrm{e}^{-t^2/2}}{\sqrt{2\pi}}\,\mathrm{d}t \quad \text{as } n \to \infty, \text{ for every } v \in \mathbb{R},$$

or, equivalently, letting $S_n := \sum_1^n X_k$,

$$\mathbb{E}f\left(S_n/\sqrt{n}\right) \to \int_{-\infty}^{\infty} f(t)\frac{\mathrm{e}^{-t^2/2}}{\sqrt{2\pi}}\,\mathrm{d}t \quad \text{as } n \to \infty, \text{ for every } f \in \mathrm{b}\mathcal{C}(\mathbb{R}).$$

Cochineal's TS1 (textcomp)

Coverage of TS1 has been enriched as of version 1.062 so that it now qualifies as sub-encoding 0, with essentially all TS1 features available.

	´0	´1	´2	´3	´4	´5	´6	´7	
´00x	0	1	2	3	4	5	6	7	˝0x
´01x	8	9	10	¸ 11	˛ 12	, 13	14	15	
´02x	| 16	¡ 17	„ 18	¿ 19	20	— 21	— 22	23	˝1x
´03x	← 24	→ 25	‿ 26	‿ 27	‿ 28	‿ 29	30	31	
´04x	ƀ 32	33	34	35	$ 36	37	38	' 39	˝2x
´05x	40	41	* 42	43	, 44	= 45	. 46	/ 47	
´06x	o 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	˝3x
´07x	8 56	9 57	Q 58	Q 59	60	− 61	62	63	
´10x	64	65	66	67	68	69	70	71	˝4x
´11x	72	73	74	75	76	Ʊ 77	78	○ 79	
´12x	80	81	82	83	84	85	86	Ω 87	˝5x
´13x	88	89	90	⟦ 91	92	⟧ 93	↑ 94	↓ 95	
´14x	96	97	★ 98	o|o 99	† 100	101	102	103	˝6x
´15x	104	105	106	107	108	ꝏ 109	♪ 110	111	
´16x	112	113	114	115	116	117	118	119	˝7x
´17x	120	121	122	123	124	125	~ 126	127	
´20x	128	129	130	131	† 132	‡ 133	‖ 134	‰ 135	˝8x
´21x	• 136	℃ 137	138	139	ƒ 140	₡ 141	₩ 142	₦ 143	
´22x	₲ 144	₱ 145	£ 146	147	₹ 148	₺ 149	₫ 150	™ 151	˝9x
´23x	‰ 152	₹ 153	₿ 154	155	℅ 156	₠ 157	∘ 158	℠ 159	
´24x	[160	] 161	¢ 162	£ 163	¤ 164	¥ 165	¦ 166	§ 167	˝Ax
´25x	168	© 169	ª 170	ⓒ 171	¬ 172	Ⓟ 173	® 174	175	
´26x	° 176	± 177	² 178	³ 179	180	µ 181	¶ 182	· 183	˝Bx
´27x	※ 184	¹ 185	º 186	√ 187	¼ 188	½ 189	¾ 190	€ 191	
´30x	192	193	194	195	196	197	198	199	˝Cx
´31x	200	201	202	203	204	205	206	207	
´32x	208	209	210	211	212	213	× 214	215	˝Dx
´33x	216	217	218	219	220	221	222	223	
´34x	224	225	226	227	228	229	230	231	˝Ex
´35x	232	233	234	235	236	237	238	239	
´36x	240	241	242	243	244	245	÷ 246	247	˝Fx
´37x	248	249	250	251	252	253	254	255	
	˝8	˝9	˝A	˝B	˝C	˝D	˝E	˝F	

Typesetting Greek with LaTeX

Cochineal offers a rather complete LGR-encoded glyph collection, lacking just a few ancient symbols.

	´0	´1	´2	´3	´4	´5	´6	´7	
´00x	– 0	̂ 1	2	3	4	5	ς 6	ς 7	˝0x
´01x	̀ 8	Ạ 9	Ḥ 10	Ọ 11	Ạ 12	Ÿ 13	α 14	ϋ 15	
´02x	́ 16	̀ 17	ɬ 18	ǫ 19	̆ 20	Ǫ 21	Ϛ 22	⅄ 23	˝1x
´03x	€ 24	‰ 25	ə 26	⅄ 27	‘ 28	’ 29	̌ 30	̄ 31	
´04x	̃ 32	! 33	̈ 34	̋ 35	̊ 36	% 37	· 38	ʼ 39	˝2x
´05x	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	
´06x	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	˝3x
´07x	8 56	9 57	: 58	· 59	ʻ 60	= 61	ʾ 62	; 63	
´10x	̃ 64	A 65	B 66	̋ 67	Δ 68	E 69	Φ 70	Γ 71	˝4x
´11x	H 72	I 73	Θ 74	K 75	Λ 76	M 77	N 78	O 79	
´12x	Π 80	X 81	P 82	Σ 83	T 84	Υ 85	˝ 86	Ω 87	˝5x
´13x	Ξ 88	Ψ 89	Z 90	[91	̃ 92	] 93	̋ 94	̊ 95	
´14x	̀ 96	α 97	β 98	ς 99	δ 100	ε 101	φ 102	γ 103	˝6x
´15x	η 104	ι 105	θ 106	κ 107	λ 108	μ 109	ν 110	ο 111	
´16x	π 112	χ 113	ρ 114	ς 115	τ 116	υ 117	118	ω 119	˝7x
´17x	ξ 120	ψ 121	ζ 122	« 123	̆ 124	» 125	̃ 126	— 127	
´20x	ὰ 128	ά 129	ὰ 130	ᾶ 131	ὰ 132	ά 133	ᾳ 134	ᾷ 135	˝8x
´21x	ά 136	ᾰ 137	ᾰ 138	ᾶ 139	ά 140	ᾱ 141	ᾱ 142	ᾷ 143	
´22x	ᾶ 144	ᾶ 145	ᾶ 146	Ϝ 147	ᾶ 148	ᾶ 149	ᾶ 150	̑ 151	˝9x
´23x	ὴ 152	ή 153	ὴ 154	155	ῆ 156	ῆ 157	ῆ 158	159	
´24x	ή 160	ῆ 161	ῆ 162	ῆ 163	ή 164	ῆ 165	ῆ 166	ῆ 167	˝Ax
´25x	ῆ 168	ῆ 169	ῆ 170	ῆ 171	ῆ 172	ῆ 173	ῆ 174	ῆ 175	
´26x	ὼ 176	ώ 177	ὼ 178	ῶ 179	ῳ 180	ῳ 181	ῳ 182	ῷ 183	˝Bx
´27x	ώ 184	ῶ 185	ῶ 186	ῶ 187	ῳ 188	ῳ 189	ῳ 190	ῷ 191	
´30x	ῶ 192	ῶ 193	ῶ 194	Ϝ 195	ῶ 196	ῶ 197	ῶ 198	199	˝Cx
´31x	ὶ 200	ί 201	ὶ 202	ῖ 203	ὺ 204	ύ 205	ὺ 206	ῦ 207	
´32x	ί 208	ΐ 209	ΐ 210	ῖ 211	ύ 212	ΰ 213	ΰ 214	ῦ 215	˝Dx
´33x	ῖ 216	ῖ 217	ῖ 218	Ϊ 219	ῦ 220	ῦ 221	ῦ 222	Ÿ 223	
´34x	ὲ 224	έ 225	ὲ 226	ῆ 227	ὸ 228	ό 229	ὸ 230	ῶ 231	˝Ex
´35x	έ 232	῅ 233	῅ 234	ῆ 235	ό 236	῅ 237	῅ 238	ῆ 239	
´36x	ϊ 240	ΐ 241	ΐ 242	ῖ 243	ϋ 244	ΰ 245	ΰ 246	ῦ 247	˝Fx
´37x	ᾳ 248	ῃ 249	ῳ 250	ῥ 251	ῥ 252	253	´ 254	¸ 255	
	˝8	˝9	˝A	˝B	˝C	˝D	˝E	˝F	

The LGR encoding defines a transcoding from Roman characters to Greek characters according to the following scheme:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Α Β ῎ Δ Ε Φ Γ Η Ι Θ Κ Λ Μ Ν Ο Π Χ Ρ Σ Τ Υ ῏ Ω Ξ Ψ Ζ
α β ς δ ε φ γ η ι ϑ ϰ λ μ ν ο π χ ρ σ τ υ ω ξ ψ ζ

The file `lgrenc.def`, which is loaded whenever `LGR` is first loaded, specifies the above and more transcodings from Latin to Greek. See the `usage.pdf` file in the `babel-greek` documentation.

The `tfm` files associated with the Cochineal `LGR` encoding were constructed with ligatures defined so that most Greek characters could be constructed using Latin input.

Note that you do not have to load the `babel` package in order for the transcoding and ligatures to function properly.

Using Cochineal LGR Greek in other LaTeX packages

There is nothing particular about Cochineal here that would not apply to any other text font package with a working LGR setup, as described above. There was for a number of years a small package `substitutefont` that allowed the substitution of, say, LGR Cochineal Greek as the LGR Greek font in another text font package that lacked a suitable one of its own. This functioned less well as LaTeX evolved and its functionality was absorbed in 2020 sby the LaTeX command `\DeclareFontFamilySubstitution`, which shares some of the problems of the now obsolete `\substitutefont`. I've added this short section to describe some work-arounds those who wish to have access to this functionality.

Here are the problems I encountered:

- If you try to add LGR Cochineal Greek to another package that has no support for LGR Greek, the attempt would fail. The work-around is to add in your preamble lines like

  ```
  \usepackage[LGR,T1]{fontenc}
  \DeclareFontFamily{LGR}{\rmdefault}{}
  ```

  after loading the text font package.

- Many text font packages do not settle on the final definition of `\rmdefault` until just before processing `\begin{document}`. You may take care of all possibilities by adding to your preamble (near the end), imagining we are adding Cochineal LGR Greek to newtx:

  ```
  \DeclareFontFamilySubstitution{LGR}{ntxtlf}{Cochineal-TLF}
  \DeclareFontFamilySubstitution{LGR}{ntxlf}{Cochineal-LF}
  \DeclareFontFamilySubstitution{LGR}{ntxtosf}{Cochineal-TOsF}
  \DeclareFontFamilySubstitution{LGR}{ntxosf}{Cochineal-OsF}
  ```

  To reconcile the difference in size between `newtx` and `Cochineal`, with respective x-heights of 450 and 432, the former is 1.042 times taller than the latter, and so we have to set the scale for Cochineal, `\Cochineal@scale` to 1.042 times that of newtx, `\ntx@scale`. So, the next line in the preamble should be something like

  ```
  \makeatletter
  \@tmpdima=\ntx@scale\p@ \@tmpdima=1.042\@tmpdima
  ```

```
\def\Cochineal@scale{\strip@pt\@tmpdima}
\makeatother
```

Typesetting Russian

With T2A encoding, the process is the same as with other T2A-encoded fonts, though the gaps in coverage may affect users of a number of non-Russian Cyrillic scripts. The only figure style provided is tabular lining (TLF.)

	´0	*´1*	*´2*	*´3*	*´4*	*´5*	*´6*	*´7*	
´00x	` 0	´ 1	ˆ 2	˜ 3	¨ 4	˝ 5	° 6	ˇ 7	˝0x
´01x	˘ 8	¯ 9	˙ 10	¸ 11	˛ 12	13	14	15	
´02x	“ 16	” 17	ˆ 18	19	˘ 20	– 21	— 22	23	˝1x
´03x	24	ı 25	ȷ 26	ff 27	fi 28	fl 29	ffi 30	ffl 31	
´04x	32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39	˝2x
´05x	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	
´06x	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	˝3x
´07x	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63	
´10x	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	˝4x
´11x	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	
´12x	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	˝5x
´13x	X 88	Y 89	Z 90	[91	\ 92	] 93	ˆ 94	_ 95	
´14x	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	˝6x
´15x	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	
´16x	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	˝7x
´17x	x 120	y 121	z 122	{ 123	| 124	} 125	~ 126	- 127	
´20x	128	129	Ђ 130	Ћ 131	132	133	134	Љ 135	˝8x
´21x	Ї 136	137	138	139	140	141	142	S 143	
´22x	144	145	Ў 146	147	148	149	Ц 150	151	˝9x
´23x	152	Є 153	154	Њ 155	Ё 156	№ 157	¤ 158	§ 159	
´24x	160	161	ђ 162	ћ 163	164	165	166	љ 167	˝Ax
´25x	ї 168	169	170	171	172	173	174	s 175	
´26x	176	177	ў 178	179	180	181	ц 182	183	˝Bx
´27x	184	є 185	ə 186	њ 187	ё 188	„ 189	« 190	» 191	
´30x	А 192	Б 193	В 194	Г 195	Д 196	Е 197	Ж 198	З 199	˝Cx
´31x	И 200	Й 201	К 202	Л 203	М 204	Н 205	О 206	П 207	
´32x	Р 208	С 209	Т 210	У 211	Ф 212	Х 213	Ц 214	Ч 215	˝Dx
´33x	Ш 216	Щ 217	Ъ 218	Ы 219	Ь 220	Э 221	Ю 222	Я 223	
´34x	а 224	б 225	в 226	г 227	д 228	е 229	ж 230	з 231	˝Ex
´35x	и 232	й 233	к 234	л 235	м 236	н 237	о 238	п 239	
´36x	р 240	с 241	т 242	у 243	ф 244	х 245	ц 246	ч 247	˝Fx
´37x	ш 248	щ 249	ъ 250	ы 251	ь 252	э 253	ю 254	я 255	
	˝8	˝9	˝A	˝B	˝C	˝D	˝E	˝F	

The OT2 encoding (supposedly obsolete, but still useful) is intended for limited use in producing Russian characters with a Western keyboard, making by means of TEX a transliteration of ASCII for most characters in the range 33–122, and providing ligatures to generate the rest, just as in the Greek case. See the documentation of `nimbus15` for further details.

	´0	*´1*	*´2*	*´3*	*´4*	*´5*	*´6*	*´7*	
´00x	Њ 0	Љ 1	Ц 2	Э 3	І 4	Є 5	Ђ 6	Ћ 7	˝0x
´01x	њ 8	љ 9	ц 10	э 11	і 12	є 13	ђ 14	ћ 15	
´02x	Ю 16	Ж 17	Й 18	Ё 19	Ѵ 20	Ѳ 21	S 22	Я 23	˝1x
´03x	ю 24	ж 25	й 26	ё 27	ѵ 28	ѳ 29	s 30	я 31	
´04x	¨ 32	! 33	” 34	Ѣ 35	˘ 36	% 37	´ 38	’ 39	˝2x
´05x	(40	) 41	* 42	ѣ 43	, 44	- 45	. 46	/ 47	
´06x	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	˝3x
´07x	8 56	9 57	: 58	; 59	« 60	ı 61	» 62	? 63	
´10x	ˇ 64	А 65	Б 66	Ц 67	Д 68	Е 69	Ф 70	Г 71	˝4x
´11x	Х 72	И 73	Ј 74	К 75	Л 76	М 77	Н 78	О 79	
´12x	П 80	Ч 81	Р 82	С 83	Т 84	У 85	В 86	Щ 87	˝5x
´13x	Ш 88	Ы 89	З 90	[91	“ 92	] 93	Ь 94	Ъ 95	
´14x	‘ 96	а 97	б 98	ц 99	д 100	е 101	ф 102	г 103	˝6x
´15x	х 104	и 105	ј 106	к 107	л 108	м 109	н 110	о 111	
´16x	п 112	ч 113	р 114	с 115	т 116	у 117	в 118	щ 119	˝7x
´17x	ш 120	ы 121	з 122	– 123	— 124	№ 125	ь 126	ъ 127	
´20x	128	129	130	131	132	133	134	135	˝8x
´21x	136	137	138	139	140	141	142	143	
´22x	144	145	146	147	148	149	150	151	˝9x
´23x	152	153	154	155	156	157	158	159	
´24x	160	161	162	163	164	165	166	167	˝Ax
´25x	168	169	170	171	172	173	174	175	
´26x	176	177	178	179	180	181	182	183	˝Bx
´27x	184	185	186	187	188	189	190	191	
´30x	192	193	194	195	196	197	198	199	˝Cx
´31x	200	201	202	203	204	205	206	207	
´32x	208	209	210	211	212	213	214	215	˝Dx
´33x	216	217	218	219	220	221	222	223	
´34x	224	225	226	227	228	229	230	231	˝Ex
´35x	232	233	234	235	236	237	238	239	
´36x	240	241	242	243	244	245	246	247	˝Fx
´37x	248	249	250	251	252	253	254	255	
	˝8	˝9	˝A	˝B	˝C	˝D	˝E	˝F	

Additional glyphs for use in German orthography

Prior to version `1.050`, `cochineal` offered basic support for German orthography, having all required accented glyphs and the lower case ß, as well as a small caps ß. Under LaTeX, the T1 encoding

contained S_S, but only as a synthesized character in the tfm. Unicode users could not make use of S_S as it was not present in the otf. So, with unicode tex processing:

```
{\addfontfeature{StylisticSet=1}\ss\ \textsc{\ss}}
```

typesets, as in LaTeX processing, to

ß ß

Note also that in unicode processing, in order to obtain the expected case change behavior, it may be necessary to add in your preamble:

```
\uccode`ß="1E9E
```

As of version 1.050 of cochineal, there are now glyphs in each style for U+1E9E and for its small caps version, as well as S_S as a real character, accessible under unicode TeX. The glyphs may be used as the uppercase and small caps versions of germandbls. Currently, the new glyphs are not available in any of the LaTeX encodings and must be used via unicode TeX.

The following tables show how to access the new glyphs in unicode TeX. Note that you will need to set StylisticSet=1 if you wish not to use the new sharp-s glyphs.

New symbols:

Glyph name	glyph	macro
uni1E9E	ẞ	\symbol{"1E9E}
uni1E9E.ss01	SS	{\addfontfeature{StylisticSet=1}\symbol{"1E9E}}
germandbls.sc	ß	{\textsc{\ss}}
germandbls.sc.ss01	ss	{\addfontfeature{StylisticSet=1}\textsc{\ss}}

Effect of choice of StylisticSet:

StylisticSet	\ss	\SS	\MakeUppercase{\ss}	\textsc{\ss}
None	ß	ẞ	ẞ	ß
=1	ß	SS	SS	ss