

The Cochineal Font Package

Michael Sharpe

November 13, 2021

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]L^AT_EX. In L^AT_EX, you access these through the macros `\textsu`, `\textinf` and `\textde`, or through their font-switching equivalents `\suf figures`, `\inf figures` and `\def figures`. For example:

- `M\textsu{lle}` Dupont and `M{\suf figures lle}` Dupont both produce M^{lle} Dupont.
- `{\inf figures 12345}` and `\textinf{12345}` render as ₁₂₃₄₅, dipping noticeably below the baseline, while `{\def figures 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 2⁶³/₆₄.

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

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
```

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 $\frac{3}{4}$.

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;
- `sup`s: 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.

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 ν 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, \dots be a sequence of iid random variables with mean 0 and variance 1 on a probability space (Ω, \mathcal{F}, P) . Then

$$P\left(\frac{X_1 + \dots + X_n}{\sqrt{n}} \leq v\right) \rightarrow N(v) := \int_{-\infty}^v \frac{e^{-t^2/2}}{\sqrt{2\pi}} dt \quad \text{as } n \rightarrow \infty,$$

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

$$Ef\left(\frac{S_n}{\sqrt{n}}\right) \rightarrow \int_{-\infty}^{\infty} f(t) \frac{e^{-t^2/2}}{\sqrt{2\pi}} dt \quad \text{as } n \rightarrow \infty, \text{ for every } f \in \mathcal{BC}(\mathbb{R}).$$

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	˘ 32	˘ 33	˘ 34	˘ 35	˘ 36	˘ 37	˘ 38	˘ 39	˘2x
´05x	˘ 40	˘ 41	˘ 42	˘ 43	˘ 44	˘ 45	˘ 46	˘ 47	
´06x	˘ 48	˘ 49	˘ 50	˘ 51	˘ 52	˘ 53	˘ 54	˘ 55	˘3x
´07x	˘ 56	˘ 57	˘ 58	˘ 59	˘ 60	˘ 61	˘ 62	˘ 63	
´10x	˘ 64	˘ 65	˘ 66	˘ 67	˘ 68	˘ 69	˘ 70	˘ 71	˘4x
´11x	˘ 72	˘ 73	˘ 74	˘ 75	˘ 76	˘ 77	˘ 78	˘ 79	
´12x	˘ 80	˘ 81	˘ 82	˘ 83	˘ 84	˘ 85	˘ 86	˘ 87	˘5x
´13x	˘ 88	˘ 89	˘ 90	˘ 91	˘ 92	˘ 93	˘ 94	˘ 95	
´14x	˘ 96	˘ 97	˘ 98	˘ 99	˘ 100	˘ 101	˘ 102	˘ 103	˘6x
´15x	˘ 104	˘ 105	˘ 106	˘ 107	˘ 108	˘ 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	˘ 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	
´32x	˘ 208	˘ 209	˘ 210	˘ 211	˘ 212	˘ 213	˘ 214	˘ 215	˘Dx
´33x	˘ 216	˘ 217	˘ 218	˘ 219	˘ 220	˘ 221	˘ 222	˘ 223	
´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.

	ó	í	¿	¿	¿	¿	ó	¿	
00x	◌ ₀	◌ ¹	◌ ₂	◌ ₃	◌ ₄	◌ ₅	Ϝ ₆	Ϝ ₇	"0x
01x	◌ ₈	Λ ₉	Η ₁₀	Ω ₁₁	Λ ₁₂	ÿ ₁₃	α ₁₄	ü ₁₅	
02x	◌ ₁₆	◌ ₁₇	η ₁₈	φ ₁₉	◌ ₂₀	Q ₂₁	Ϝ ₂₂	Ϸ ₂₃	"1x
03x	€ ₂₄	% ₂₅	ə ₂₆	Ϸ ₂₇	◌ ₂₈	◌ ₂₉	◌ ₃₀	◌ 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	Α ₆₅	Β ₆₆	◌ ₆₇	Δ ₆₈	Ε ₆₉	Φ ₇₀	Γ ₇₁	"4x
11x	Η ₇₂	Ι ₇₃	Θ ₇₄	Κ ₇₅	Λ ₇₆	Μ ₇₇	Ν ₇₈	Ο ₇₉	
12x	Π ₈₀	Χ ₈₁	Ρ ₈₂	Σ ₈₃	Τ ₈₄	Υ ₈₅	◌ ₈₆	Ω ₈₇	"5x
13x	Ξ ₈₈	Ψ ₈₉	Ζ 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	ά ₁₂₈	ά ₁₂₉	ά ₁₃₀	ά ₁₃₁	ά ₁₃₂	ά ₁₃₃	ά ₁₃₄	ά ₁₃₅	"8x
21x	ά ₁₃₆	ά ₁₃₇	ά ₁₃₈	ά ₁₃₉	ά ₁₄₀	ά ₁₄₁	ά ₁₄₂	ά ₁₄₃	
22x	ά ₁₄₄	ά ₁₄₅	ά ₁₄₆	Ϝ ₁₄₇	ά ₁₄₈	ά ₁₄₉	ά ₁₅₀	◌ ₁₅₁	"9x
23x	ή ₁₅₂	ή ₁₅₃	ή ₁₅₄	◌ ₁₅₅	ή ₁₅₆	ή ₁₅₇	ή ₁₅₈	◌ ₁₅₉	
24x	ή ₁₆₀	ή ₁₆₁	ή ₁₆₂	ή ₁₆₃	ή ₁₆₄	ή ₁₆₅	ή ₁₆₆	ή ₁₆₇	"Ax
25x	ή ₁₆₈	ή ₁₆₉	ή ₁₇₀	ή ₁₇₁	ή ₁₇₂	ή ₁₇₃	ή ₁₇₄	ή ₁₇₅	
26x	ώ ₁₇₆	ώ ₁₇₇	ώ ₁₇₈	ώ ₁₇₉	ώ ₁₈₀	ώ ₁₈₁	ώ ₁₈₂	ώ ₁₈₃	"Bx
27x	ώ ₁₈₄	ώ ₁₈₅	ώ ₁₈₆	ώ ₁₈₇	ώ ₁₈₈	ώ ₁₈₉	ώ ₁₉₀	ώ ₁₉₁	
30x	ώ ₁₉₂	ώ ₁₉₃	ώ ₁₉₄	Ϝ ₁₉₅	ώ ₁₉₆	ώ ₁₉₇	ώ ₁₉₈	◌ ₁₉₉	"Cx
31x	ì ₂₀₀	ì ₂₀₁	ì ₂₀₂	ì ₂₀₃	ù ₂₀₄	ù ₂₀₅	ù ₂₀₆	ù ₂₀₇	
32x	ì ₂₀₈	ì ₂₀₉	ì ₂₁₀	ì ₂₁₁	ú ₂₁₂	ÿ ₂₁₃	ÿ ₂₁₄	ÿ ₂₁₅	"Dx
33x	ĩ ₂₁₆	ĩ ₂₁₇	ĩ ₂₁₈	ĩ ₂₁₉	ÿ ₂₂₀	ÿ ₂₂₁	ÿ ₂₂₂	ÿ ₂₂₃	
34x	è ₂₂₄	é ₂₂₅	é ₂₂₆	é ₂₂₇	ò ₂₂₈	ó ₂₂₉	ò ₂₃₀	ò ₂₃₁	"Ex
35x	é ₂₃₂	é ₂₃₃	é ₂₃₄	é ₂₃₅	ó ₂₃₆	ó ₂₃₇	ó ₂₃₈	ó ₂₃₉	
36x	ï ₂₄₀	ï ₂₄₁	ï ₂₄₂	ï ₂₄₃	ÿ ₂₄₄	ÿ ₂₄₅	ÿ ₂₄₆	ÿ ₂₄₇	"Fx
37x	α ₂₄₈	η ₂₄₉	ω ₂₅₀	ρ ₂₅₁	ρ ₂₅₂	◌ ₂₅₃	◌ ₂₅₄	◌ ₂₅₅	
	"8	"9	"A	"B	"C	"D	"E	"F	

Setting up your source file to use LGR usually involves `babel` and `substitutefont`. See the documentation of `nimbus15` for details.

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.)

	ó	í	ú	û	ü	ý	ö	ÿ	
´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	1 25	J 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	^ 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	Ђ 131	132	133	134	Љ 135	"8x
´21x	Ї 136	137	138	139	140	141	142	Ѕ 143	
´22x	144	145	Ў 146	147	148	149	Ц 150	151	"9x
´23x	152	Є 153	154	Ѓ 155	Ё 156	№ 157	Ѡ 158	Ѕ 159	
´24x	160	161	ђ 162	ђ 163	164	165	166	Љ 167	"Ax
´25x	ї 168	169	170	171	172	173	174	Ѕ 175	
´26x	176	177	ў 178	179	180	181	Ц 182	183	"Bx
´27x	184	Є 185	Ә 186	Ђ 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 T_EX a transliteration of ASCII for most characters in the range 33–122, and providing ligatures to generate the rest. See the documentation of nimbus15 for further details.

	ó	í	ú	ÿ	´	¸	¸	¸	
ó0x	Ъ ₀	Ь ₁	Ц ₂	Э ₃	І ₄	Є ₅	Ђ ₆	Ћ ₇	"0x
ó1x	ъ ₈	ь ₉	ц ₁₀	э ₁₁	і ₁₂	є ₁₃	ђ ₁₄	ћ ₁₅	
ó2x	Ю ₁₆	Ж ₁₇	Й ₁₈	Ё ₁₉	Ѹ ₂₀	Ө ₂₁	Š ₂₂	Я ₂₃	"1x
ó3x	ю ₂₄	ж ₂₅	й ₂₆	ё ₂₇	ѹ ₂₈	ө ₂₉	š ₃₀	я ₃₁	
ó4x	¨ ₃₂	! ₃₃	" ₃₄	Ђ ₃₅	ˇ ₃₆	‰ ₃₇	´ ₃₈	’ ₃₉	"2x
ó5x	(₄₀)) ₄₁	* ₄₂	ђ ₄₃	, ₄₄	- ₄₅	· ₄₆	/ ₄₇	
ó6x	0 ₄₈	1 ₄₉	2 ₅₀	3 ₅₁	4 ₅₂	5 ₅₃	6 ₅₄	7 ₅₅	"3x
ó7x	8 ₅₆	9 ₅₇	: ₅₈	; ₅₉	« ₆₀	! ₆₁	» ₆₂	? ₆₃	
ó8x	˘ ₆₄	А ₆₅	Б ₆₆	Ц ₆₇	Д ₆₈	Е ₆₉	Ф ₇₀	Г ₇₁	"4x
ó9x	Х ₇₂	И ₇₃	Ј ₇₄	К ₇₅	Л ₇₆	М ₇₇	Н ₇₈	О ₇₉	
óax	П ₈₀	Ч ₈₁	Р ₈₂	С ₈₃	Т ₈₄	У ₈₅	В ₈₆	Щ ₈₇	"5x
óbx	Ш ₈₈	Ы ₈₉	Э ₉₀	[₉₁	" ₉₂] ₉₃	Ь ₉₄	Ђ ₉₅	
ócx	‘ ₉₆	а ₉₇	б ₉₈	ц ₉₉	д ₁₀₀	е ₁₀₁	ф ₁₀₂	г ₁₀₃	"6x
ódx	х ₁₀₄	и ₁₀₅	ј ₁₀₆	к ₁₀₇	л ₁₀₈	м ₁₀₉	н ₁₁₀	о ₁₁₁	
óex	п ₁₁₂	ч ₁₁₃	р ₁₁₄	с ₁₁₅	т ₁₁₆	у ₁₁₇	в ₁₁₈	щ ₁₁₉	"7x
ófx	ш ₁₂₀	ы ₁₂₁	э ₁₂₂	– ₁₂₃	— ₁₂₄	№ ₁₂₅	ь ₁₂₆	ђ ₁₂₇	
	"8	"9	"A	"B	"C	"D	"E	"F	

Running cochineal in XeLaTeX or LuaLaTeX

The preceding pages of this document were processed using pdf \LaTeX , which was the only engine supported by cochineal.sty. As of version 1.07, XeLaTeX and LuaLaTeX are also supported. All cochineal options and macros formerly limited to pdf \LaTeX have been modified and now work essentially the same under both unicode engines, though the output may not always be precisely the same as with pdf \LaTeX . There are some macros and options available under unicode engines that go beyond what can be done under pdf \LaTeX .

MODIFIED MACROS AND OPTIONS:

- `\textfrac` works better under unicode tex because it is possible to adjust kerning between all characters. The effect should not be very noticeable, at least in regular style.
- `sup`s: the package treats this differently in unicode LaTeX and pdf \LaTeX , with handling in the unicode case passed off to the `realscripts` package where the footnote marker font is set to `\normalfont`, meaning that superiors from the current (Cochineal) text font are employed.
- `theoremfont`, `thmtabular`, `thmlining` all function in a similar manner to pdf \LaTeX .
- `swashQ` operates as before.
- `foresolidus`, `aftsolidus` are not used in unicode tex.
- `scosf` operates without change.
- `\textcircled` works the same as under pdf \LaTeX .[Ⓢ]

NEW MACROS AND OPTIONS:

- An alternate version of Q, Q (`\Qswash`) and their small cap versions is available using option `altQ`, which sets `StylisticSet=3`. The alternate versions are \mathring{Q} , \mathring{Q} .
- An alternate version of J is available in italic shapes only using option `altJ`, which sets `StylisticSet=2`. The alternate version of *J* is *J*.
- `oldSS` controls whether the new German capital sharp S is used or whether the old SS is retained. The former is the default but the option `oldSS` forces the latter by setting `StylisticSet=1`. The effects are summarized in the following tables.

Glyph name	glyph	macro
uni1E9E	Œ	<code>\symbol{"1E9E}</code> or <code>\SS</code>
uni1E9E.ss01	SS	<code>{\addfontfeature{StylisticSet=1}\symbol{"1E9E}}</code>
germandbls.sc	ſ	<code>{\textsc{\ss}}</code>
germandbls.sc.ss01	ss	<code>{\addfontfeature{StylisticSet=1}\textsc{\ss}}</code>

Effect of choice of `StylisticSet`:

StylisticSet	<code>\ss</code>	<code>\SS</code>	<code>\MakeUppercase{\ss}</code>	<code>\textsc{\ss}</code>
None	ſ	Œ	Œ	ſ
=1	ſ	SS	SS	ss