

## Greek alphabet

Greek letters via Latin transscription in LGR font encoding:

A B Γ Δ E Z H Θ I K Λ M N Ξ O Π P Σ T Υ Φ X Ψ Ω

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω

Greek letters via default macros in other font encoding (here T1):

A B Γ Δ E Z H Θ I K Λ M N Ξ O Π P Σ T Υ Φ X Ψ Ω

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω

So you can easily write a single Greek symbol (like Ψ or μ) or λογος in non-Greek text or ISO-conforming formulas with an upright pi symbol like  $A = \pi r^2$  instead of  $A = \pi r^2$ .

However, kerning is broken in other font encodings: compare ATA (LGR) to ATA(T1). Because of this (and for proper hyphenation), use of babel and correct language setting is recommended for Greek quotes.

## λογος, λογος and λογος

The subsection title above uses: text\* macros, Unicode input and the LGR transcription for the Greek word λογος. The last variant (Latin transscription) is written in Latin letters in the PDF bookmark.

On 2010-11-05, Heiko Oberdiek wrote in comp.text.tex:

```
\pdfstringdef (converting TeX code to PDF strings for bookmarks)
supports NFSS2 and needs active characters. Encoding stuff based
on the internal font machinery of TeX (letters with catcode 11 or
12, ligatures) does not work, because the strings don't reach TeX's
stomach.
```

## Diacritics

In LGR encoding, shortcuts as well as named macros are set up for use with the Latin transcription.

Composition of diacritics (like \Dasia\Tonos) fails in other font encodings. Long names (like \DasiaOxia) work, however they do not select precomposed characters (the difference becomes obvious if you drag-and-drop text from the PDF version of this document): LGR: ᾰ ᾱ ᾲ ᾳ

Latin: ǎ

Diacritics (except the dialytika) should be placed before capital letters and dropped with MakeUppercase:

$\acute{\alpha}$   $\acute{\epsilon}$   $\acute{\iota}$   $\grave{\eta}$   $\acute{\omicron}$   $\acute{\upsilon}$   $\acute{\omega}$   
 $\grave{\alpha}$   $\grave{\epsilon}$   $\grave{\iota}$   $\grave{\eta}$   $\grave{\omicron}$   $\grave{\upsilon}$   $\grave{\omega}$   
 $\acute{\alpha}$   $\acute{\epsilon}$   $\acute{\iota}$   $\acute{\eta}$   $\acute{\omicron}$   $\acute{\upsilon}$   $\acute{\omega}$

However, in other font encodings, this does not work:

No kerning: ( $\acute{\alpha}\acute{\alpha}$  vs.  $\acute{\alpha}\acute{\alpha}$ ), misplaced multiple ( $\acute{\alpha}$  vs.  $\acute{\alpha}$ ) or Titlecase accents ( $\acute{\alpha}$   $\acute{\epsilon}$   $\acute{\iota}$   $\acute{\eta}$   $\acute{\omicron}$   $\acute{\upsilon}$   $\acute{\omega}$  vs.  $\acute{\alpha}$   $\acute{\epsilon}$   $\acute{\iota}$   $\acute{\eta}$   $\acute{\omicron}$   $\acute{\upsilon}$   $\acute{\omega}$ ) and errors with MakeUppercase.

Therefore, we would need to define macros for all Unicode characters that we want to access without explicit font-encoding change.

Alternative: A `\TextGreek` TextCommandDefault that wraps the argument in the equivalent of Babel's `\textgreek` with an LGR TextCommand that just passes the argument. This is done in the preamble of this document.

This works with the Babel transscription ( $\lambda\omicron\gamma\omicron\varsigma$ ), keeps kerning (if the kerning pair is inside the argument,  $\acute{\alpha}\acute{\alpha}$ ), and allows iterative accent macros where pre-composed characters are selected ( $\acute{\alpha}$  or  $\acute{\alpha}$ ).

TODO: Use this definition for multi-macro definitions in Unicode definition (\*.udf) files?