The Next 700 Classical Languages

Kyle P. Johnson

Principal Research Scientist
Accenture, Operations AI

Future Philologies: Digital Directions in Ancient World Text
April 20, 2018

Outline

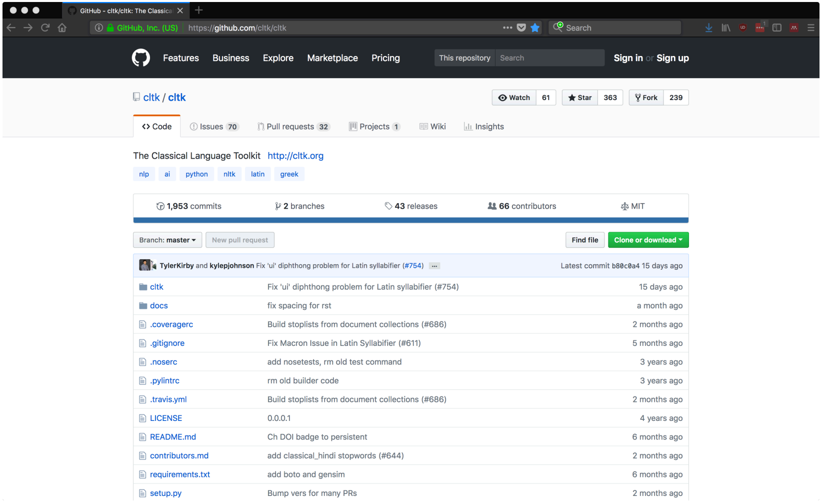
“The languages people use to communicate with computers differ in their intended aptitudes, towards either a particular application area, or a particular phase of computer use .... They also differ in physical appearance, and more important, in logical structure. The question arises, do the idiosyncracies reflect basic logical properties of the situations that are being catered for? **Or are they accidents of history and personal background that may be obscuring fruitful developments?** ...”

“… This question is clearly important if we are **trying to predict or influence language evolution**. To answer it we must think in terms, not of languages, but of families of languages. That is to say we must systematize their design **so that a new language is a point chosen from a well-mapped space, rather than a laboriously devised construction**.” (Peter Landin, “The Next 700 Programming Languages” 1966)

CLTK

At a glance

- Software for natural language processing (NLP) in languages of Ancient, Classical, and Medieval Eurasia
- Free and open source (MIT)
- Started in 2012
- 30 languages; 66 contributors; 2,000 commits, almost 100 releases
- Board of Advisors
- Raised grants to support 7 students' summer stipends
- But money not required
- corpora and infrastructure to support adoption of NLP among ancient–world researchers



The Classical Language Toolkit http://cltk.org

nlp ai python nltk latin greek

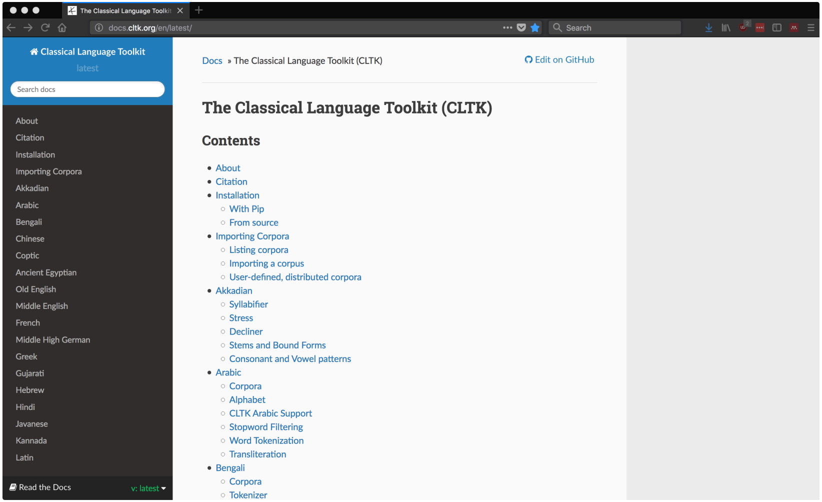
1,953 commits · 2 branches · 43 releases · 66 contributors · MIT

Branch: master · New pull request · Find file · Clone or download

TylerKirby and kylepjohnson Fix 'ui' diphthong problem for Latin syllabifier (#754) … Latest commit b80c0a4 15 days ago

cltk	Fix 'ui' diphthong problem for Latin syllabifier (#754)	15 days ago
docs	fix spacing for rst	a month ago
.coveragerc	Build stoplists from document collections (#686)	2 months ago
.gitignore	Fix Macron Issue in Latin Syllabifier (#611)	5 months ago
.noserc	add nosetests, rm old test command	3 years ago
.pylintrc	rm old builder code	3 years ago
.travis.yml	Build stoplists from document collections (#686)	2 months ago
LICENSE	0.0.0.1	4 years ago
README.md	Ch DOI badge to persistent	6 months ago
contributors.md	add classical_hindi stopwords (#644)	2 months ago
requirements.txt	add boto and gensim	6 months ago
setup.py	Bump vers for many PRs	2 months ago

Figure: GitHub core project page (source: “GitHub: cltk/cltk”)



Classical Language Toolkit
latest
Search docs

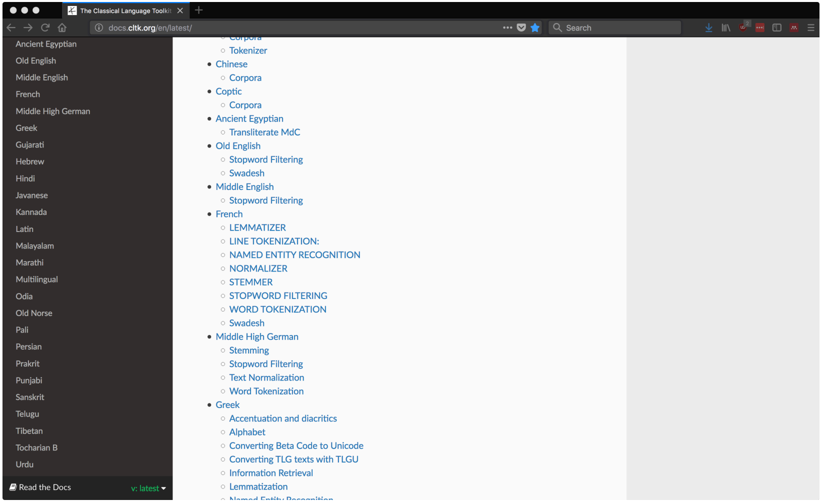
Read the Docs v: latest

Docs » The Classical Language Toolkit (CLTK) Edit on GitHub

The Classical Language Toolkit (CLTK)

Contents

Figure: Docs core project page (source: “Docs: The Classical Language Toolkit”)


The Classical Language Toolkit
docs.cltk.org/en/latest/
Search

Ancient Egyptian
Old English
Middle English
French
Middle High German
Greek
Gujarati
Hebrew
Hindi
Javanese
Kannada
Latin
Malayalam
Marathi
Multilingual
Odia
Old Norse
Pali
Persian
Prakrit
Punjabi
Sanskrit
Telugu
Tibetan
Tocharian B
Urdu

Read the Docs v: latest

- Corpora
- Tokenizer
- Chinese
  - Corpora
- Coptic
  - Corpora
- Ancient Egyptian
  - Transliterate MdC
- Old English
  - Stopword Filtering
  - Swadesh
- Middle English
  - Stopword Filtering
- French
  - LEMMATIZER
  - LINE TOKENIZATION:
  - NAMED ENTITY RECOGNITION
  - NORMALIZER
  - STEMMER
  - STOPWORD FILTERING
  - WORD TOKENIZATION
  - Swadesh
- Middle High German
  - Stemming
  - Stopword Filtering
  - Text Normalization
  - Word Tokenization
- Greek
  - Accentuation and diacritics
  - Alphabet
  - Converting Beta Code to Unicode
  - Converting TLG texts with TLGU
  - Information Retrieval
  - Lemmatization
  - Named Entity Recognition

Figure: Docs core project page (source: “Docs: The Classical Language Toolkit”)

Three areas of digital Classics

- Corpora
- Software
- Infrastructure

Three areas of digital Classics

- Corpora ≈ *what*
- Software ≈ *how*
- Infrastructure ≈ *where*

Three areas of digital Classics

- Corpora ≈ *extract*
- Software ≈ *transform*
- Infrastructure ≈ *load*

Three areas of digital Classics

- Corpora ≈ texts, translations, dictionaries, grammars, textbooks
- Software ≈ methodologies, disciplines, theories
- Infrastructure ≈ Universities, colleges, departments, institutes, societies, journals, publishers, colloquia

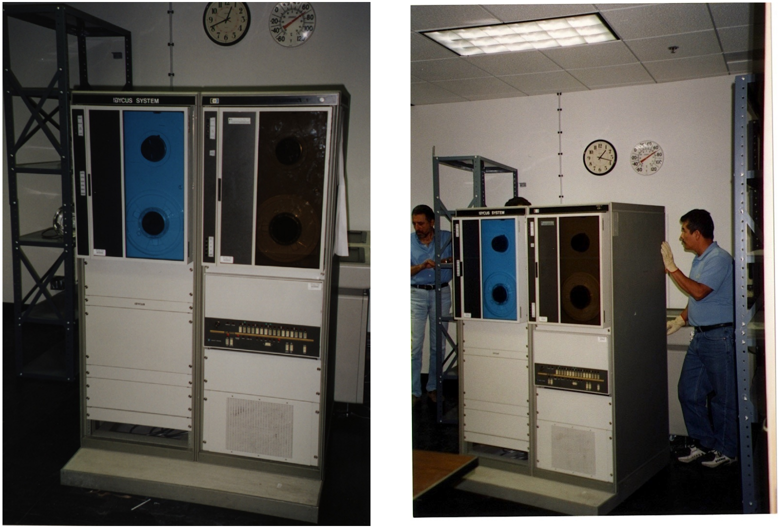


Figure: Ibycus: David Packard's modified HP-1000 (source: "Ibycus")

Corpora

Brief history

- Big projects
  - Encoding and representation: Beta Code, GreekKeys, UTF-16, UTF-8
  - Texts: Packard's concordance to Livy, TLG, PHI, LacusCurtius, The Latin Library, Perseus, Papyri.info, Trismegistos, Suda On Line, Alpheios Project, Homer Multitext, Open Greek and Latin
  - Treebanks, etc.: Index Thomisticus, Alpheos, PROIEL
  - Scholarship: Diotima, L'Année philologique on the Internet
- Logical properties
  - Text input: Expensive, major projects with institutional backing
  - Encoding and representation: Convergence on Unicode and UTF-8
  - Markup: Several major families of standards (TLG & PHI), TEI XML
  - Citation: CITE URN, still emerging
  - Licensing: Movement toward public domain texts, rereleased under FOSS
  - Version control: Recent adoption (e.g., Open Greek and Latin)

Corpora

Trying to predict

- Encoding and representation: solved (Unicode and UTF-8)
  - However Unicode Consortium's work not finished (Egyptian hieroglyphics, Mayan)
- Markup: not solved. Why?
  - Expense of input and labelling work
  - Some standards encourage complexity; complexity leads to incompatibility
  - XML is probably best, however difficult for humans to read and edit
- Licensing: Public domain/FOSS texts more widely adopted
  - "open access" (like PHI Online [http://latin.packhum.org/browse] have had less impact
- Version control: Of course

Corpora

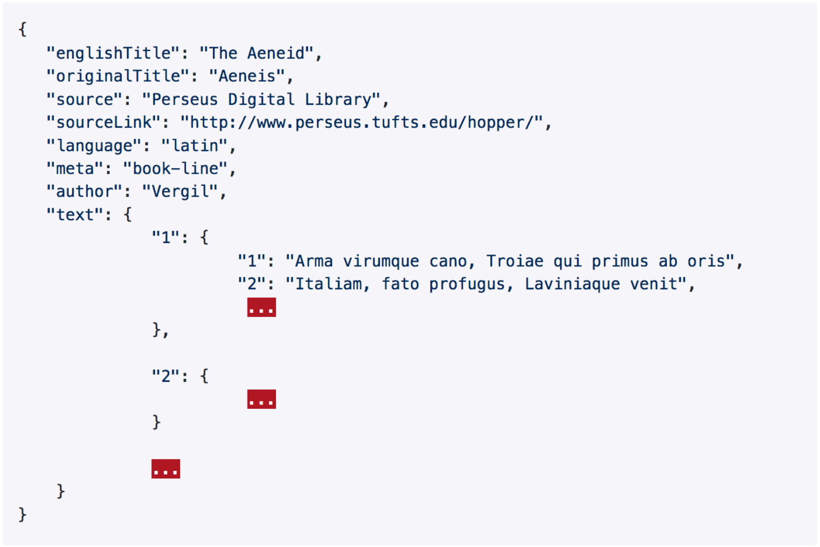
CLTK: Fruitful developments?

- Encoding and representation: Python 3 for native Unicode support
- Markup: Parsers for major standards (PHI/TLG and TEI XML)
  - Come as you are: However most texts have no encoding (WikiSource) or an uncommon one (CBETA)
  - Middle ground: CLTK JSON object
- Licensing: We can only serve public domain texts
- Version control: We host all texts in Git, with each edit to a text creating a new version ("Git hash")
  - For projects already in Git, we "fork" a version of the official repo, then manage upstream updates
- Weaknesses
  - Citation: Common interface for accessing passages
  - Markup: Tension between NLP and philology
  - Markup: Low adoption for CLTK JSON object

Corpora

CLTK: Fruitful developments?

- Encoding and representation: Python 3 for native Unicode support
- Markup: Parsers for major standards (PHI/TLG and TEI XML)
  - Come as you are: However most texts have no encoding (WikiSource) or an uncommon one (CBETA)
  - Middle ground: CLTK JSON object
- Licensing: We can only serve public domain texts
- Version control: We host all texts in Git, with each edit to a text creating a new version ("Git hash")
  - For projects already in Git, we "fork" a version of the official repo, then manage upstream updates
- Weaknesses
  - Citation: Common interface for accessing passages
  - Markup: Tension between NLP and philology
  - Markup: Low adoption for CLTK JSON object



```
{
   "englishTitle": "The Aeneid",
   "originalTitle": "Aeneis",
   "source": "Perseus Digital Library",
   "sourceLink": "http://www.perseus.tufts.edu/hopper/",
   "language": "latin",
   "meta": "book-line",
   "author": "Vergil",
   "text": {
              "1": {
                       "1": "Arma virumque cano, Troiae qui primus ab oris",
                       "2": "Italiam, fato profugus, Laviniaque venit",
                         ...
              },

              "2": {
                         ...
              }

              ...
    }
}
```

Figure: CLTK's JSON object (source: "CLTK API Wiki")

Software

Brief history

- Big projects
  - Whitaker's Words, Morpheus, Perseus, Alphaeus, Perseus under PhiloLogic, Tesserae, Scaife Viewer
  - TLG readers: Ibycus, Antiquarium, Libycus, TLG Workplace, Offload, Chiron, Lector, V&F, Musaios, SNS Greek and Latin, Accordance, Hellespont, Lexis, TLG Engine, Pandora, Diogenes
  - Scripts on GitHub: Many of these, recent
- Logical properties
  - Data: applications proliferate under standardized corpus markup (TLG/PHI, Perseus TEI XML)
  - Programming languages: Many
  - Location: From local to web applications
  - Licensing: Rise of FOSS, wider adoption
  - Version control: Increased adoption

Software

Trying to predict

- Languages: Programming language must handle Unicode
- Development: Applications should be more modular, less costly to extend
- Algorithms: NLP is specialized, tool–making should be offloaded to specialists
- Licensing: FOSS software and algorithms essential for adoption
- Version control: Of course

Software

CLTK: Fruitful developments?

- Algorithms: NLP algorithms exclusively designed for reuse
- Development: Code kept as simple as possible, to increase interpretability and hack–ability
- Scholarship: Use scholarship–vetted and best–practice methods
- Coverage: Breadth over depth
- Scientific method: Common (and versions) algorithms for reproduction of results
- Weaknesses
    - Can statistical NLP meet the precision expected by philologists?
    - Few documented algorithms for classical languages; when to innovate?
    - Depth over breadth?
    - Web projects are challenging

Software

CLTK: Fruitful developments?

- Algorithms: NLP algorithms exclusively designed for reuse
- Development: Code kept as simple as possible, to increase interpretability and hack–ability
- Scholarship: Use scholarship–vetted and best–practice methods
- Coverage: Breadth over depth
- Scientific method: Common (and versions) algorithms for reproduction of results
- Weaknesses
  - Can statistical NLP meet the precision expected by philologists?
  - Few documented algorithms for classical languages; when to innovate?
  - Depth over breadth?
  - Web projects are challenging

Infrastructure

Brief history

- Big projects
  - Individuals: Many (e.g, Whitaker's words)
  - Small teams: Some (e.g, Alphaeos)
  - Large teams: A few (e.g, TLG, Perseus)
  - Crowd sourced: One or two (e.g, Suda On Line, POS voting in Perseus)
- Logical properties
  - Collaboration: Development teams distributed, but contributors reflect real–world groups
  - Collaboration: Not necessarily closed, but not designed for openness
  - Data distribution: tape, floppy disks, CD-ROM, Internet

Infrastructure

Trying to predict

- Decentralization: General trend (e.g., WWW to P2P)
- Scientific method: Digital humanities are quantitative; reproduction of experiments required
- Funding: Expected decrease of usual funding sources in ancient language humanities (NEH, departments, societies)
- Purpose: Funding usually aimed at producing stand–alone projects
- Knowledge: Expected decrease of experts knowledgeable in ancient languages (globalization, modernization, loss of students)

Infrastructure

CLTK: Fruitful developments?

- Decentralization data: Any Git–backed corpus or software can be imported into a user's environment
- Decentralized software: Permissive FOSS license
- Decentralized groups: Contribs are networked across boundaries of discipline, nation, etc.
- Smart centralization: A team of admins, overseen by Board of Advisors, safeguarding core software
- Purpose: Fills the gap of reusable code libraries, not traditionally rewarded by scholarly economy
- Economy: Humanists compete for grants is a zero-sum game CLTK does not
- Weaknesses
    - Modular communities: Our goal is self–organizing communities of experts
    - Resources: Endangered and low–resource languages

Infrastructure

CLTK: Fruitful developments?

- Decentralization data: Any Git–backed corpus or software can be imported into a user's environment
- Decentralized software: Permissive FOSS license
- Decentralized groups: Contribs are networked across boundaries of discipline, nation, etc.
- Smart centralization: A team of admins, overseen by Board of Advisors, safeguarding core software
- Purpose: Fills the gap of reusable code libraries, not traditionally rewarded by scholarly economy
- Economy: Humanists compete for grants is a zero-sum game CLTK does not
- Weaknesses
  - Modular communities: Our goal is self–organizing communities of experts
  - Resources: Endangered and low–resource languages

“The question arises, do the idiosyncracies reflect basic logical properties of the situations that are being catered for? **Or are they accidents of history and personal background that may be obscuring fruitful developments?** This question is clearly important if we are **trying to predict or influence language evolution**. To answer it we must think in terms, not of languages, but of families of languages. That is to say we must systematize their design **so that a new language is a point chosen from a well-mapped space, rather than a laboriously devised construction**.”