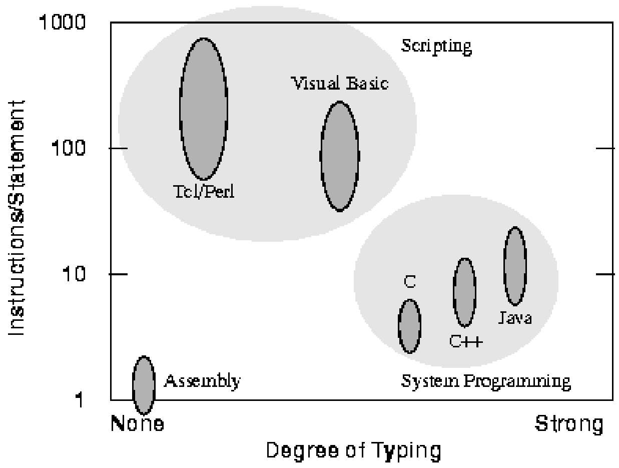
scripting: References

- John Ousterhout’s IEEE article
  “Scripting: Higher Level Programming for the 21st Century”
  `http://www.scriptics.com/people/john.ousterhout/scripting.html`
- Guido Van Rossum’s CNRI proposal
  “Computer Programming for Everybody”
  `http://www.python.org/doc/essays/cp4e.html`
- Python starting point
  `http://www.python.org/`
- Tcl/Tk starting point
  `http://www.scriptics.com/`

scripting: language classification



scripting: Why ?

- purpose: glueing, system integration, *complimentary* to system programming languages. Choose paradigm $\rightarrow$ choose language $\rightarrow$ solve problem.
- examples: command shells (sh, csh, zsh, . . .), Perl, Tcl, Python, Visual Basic, . . .
- Rapid Application Development (RAD):
  - interpreted (no edit/compile/link/test/. . .)
  - weak typing, but verbose and safe error diagnostics
  - garbage collected
  - powerful basic structures (control, dictionaries, exceptions, . . .)
- glue through extensions

- embedding
- incremental development: dynamic loading (.so, .dll)
- platform-independent (Mac, Windows, UNIX)
- plethora of existing extensions (Tcl and Python)

Python

Reference book: Mark Lutz, “Programming Python”, O’Reilly

- basic
- exceptions, classes
- Tk (Tkinter)

  `http://www.pythonware.com/library/tkinter/introduction/index.htm`
- process, thread, . . .
- extensions (for TESTING)
- embedding (for user customisation)
- JPython
- Design Patterns in Python

  `http://www.python.org/workshops/1997-10/proceedings/savikko.html`