

quarto-index sort-key probe

Terms that file under something other than what they print

Every term below is written so that its sorted position differs from the position its printed text alone would give it. The order the marks appear in is deliberately not the order the index must print, and the twin fixture — this file with every `sort=` attribute removed and nothing else changed — prints them in a demonstrably different order.

A leading article files under the word that matters: The Hague.

A name whose first character sorts nowhere near where a reader looks for it: Ångström.

A number files where it is spoken: 10 Downing Street.

A nobiliary particle is not the surname: von Neumann.

A surname files ahead of the given name that precedes it: Édouard Manet.

A second name filed by its surname, whose printed and sorted positions are chosen so that the two orders of this fixture share no position at all — without it two of these terms would happen to land in the same slot under both orders, and the fixture could no longer tell a sorted index from an unsorted one: Ursula K. Le Guin.

One sort key, several marks

The same term marked again, this time with no `sort=` of its own. A sort key belongs to the entry rather than to the mark that happened to declare it, so this mark files under **Hague** too — were it to file under its printed text instead, the index would carry the term twice, in two places, printed identically: The Hague.

Sub-entries sort by their own keys

Two sub-entries under one parent, written in the order their printed text would give them, so the index cannot be right by having left them alone. The first level of each sort key is empty, which means "file this level under its own printed text" — only the second level is being redirected.

Turing: Neumann:

Index

Ångström, [1](#)

The Hague, [1](#)

Ursula K. Le Guin, [1](#)

Édouard Manet, [1](#)

mathematicians

von Neumann, [2](#)

Alan Turing, [2](#)

von Neumann, [1](#)

10 Downing Street, [1](#)