Analysis

- Sometimes referred to as *text analysis*
- Applicable to text fields/values
- Text values are analyzed when indexing documents
- The result is stored in data structures that are efficient for searching etc.
- The `_source` object is **not** used when searching for documents
    - It contains the exact values specified when indexing a document

CodingExplained
CODE. SLEEP. REPEAT.

Character filters

- Adds, removes, or changes characters
- Analyzers contain zero or more character filters
- Character filters are applied in the order in which they are specified
- Example (`html_strip` filter)
  - **Input:** `"I&apos;m in a <em>good</em> mood - and I <strong>love</strong> açaí!"`
  - **Output:** `"I'm in a good mood - and I love açaí!"`

CodingExplained
CODE. SLEEP. REPEAT.

Tokenizers

- An analyzer contains **one** tokenizer
- Tokenizes a string, i.e. splits it into tokens
- Characters may be stripped as part of the tokenization
- Example
  - **Input:** `"I REALLY like beer!"`
  - **Output:** `["I", "REALLY", "like", "beer"]`

CodingExplained
CODE. SLEEP. REPEAT.

Token filters

- Receive the output of the tokenizer as input (i.e. the tokens)
- A token filter can add, remove, or modify tokens
- An analyzer contains zero or more token filters
- Token filters are applied in the order in which they are specified
- Example (`lowercase` filter)
  - **Input:** `["I", "REALLY", "like", "beer"]`
  - **Output:** `["i", "really", "like", "beer"]`

CodingExplained
CODE. SLEEP. REPEAT.

Built-in and custom components

- Built-in analyzers, character filters, tokenizers, and token filters are available
- We can also build custom ones
    - You will see how later in this section
- For now, let’s see how text values are analyzed by default...

CodingExplained
CODE. SLEEP. REPEAT.