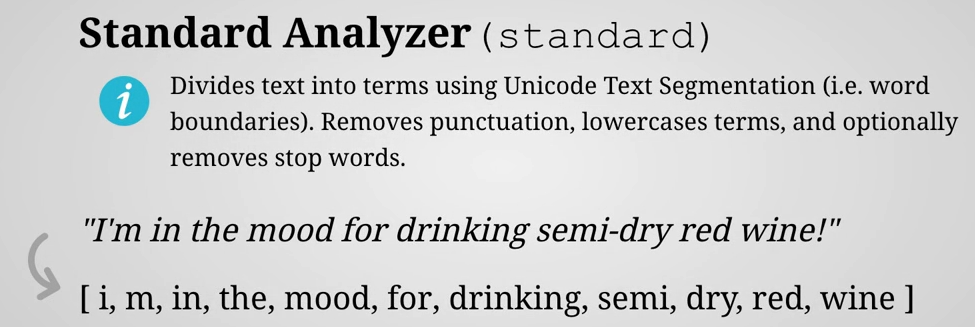
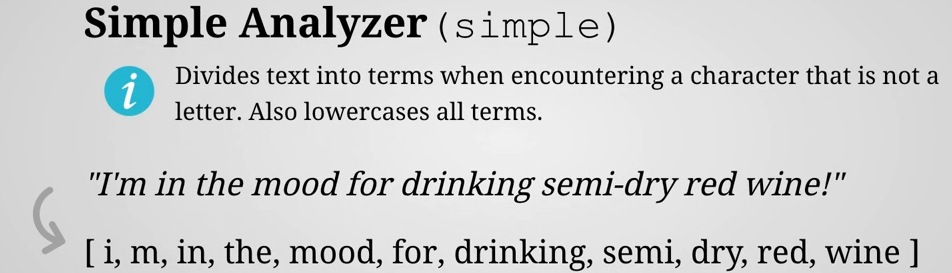
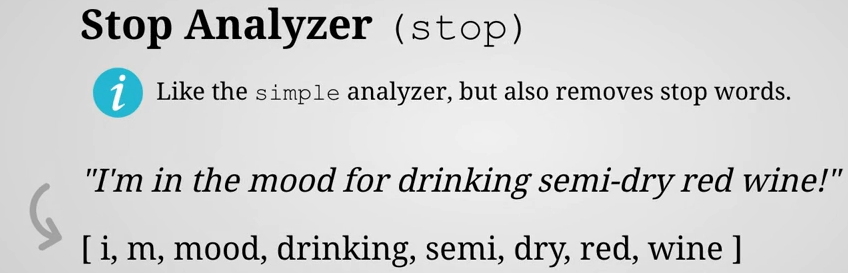
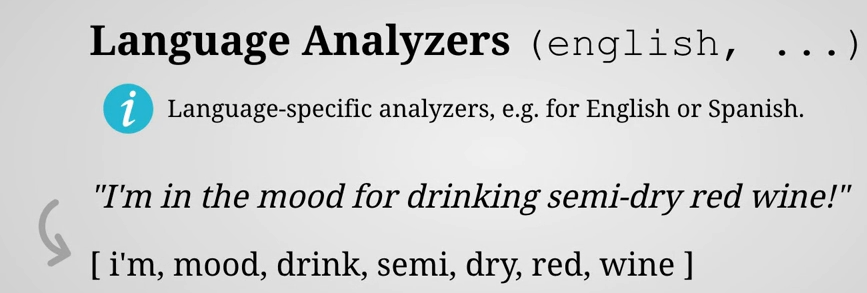
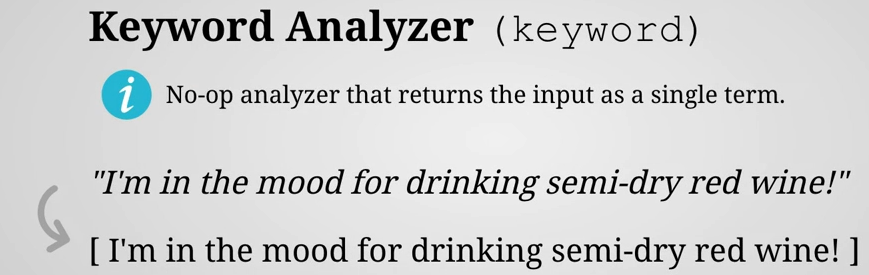
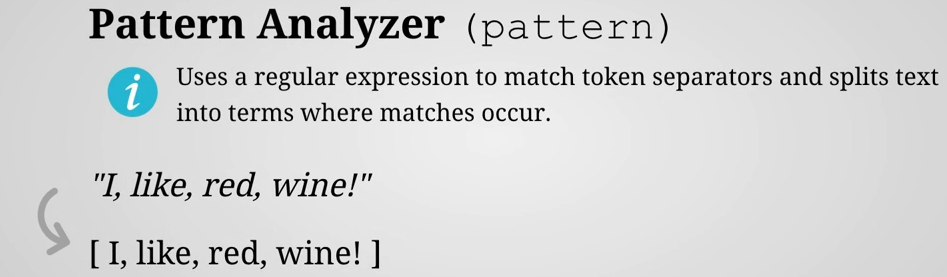
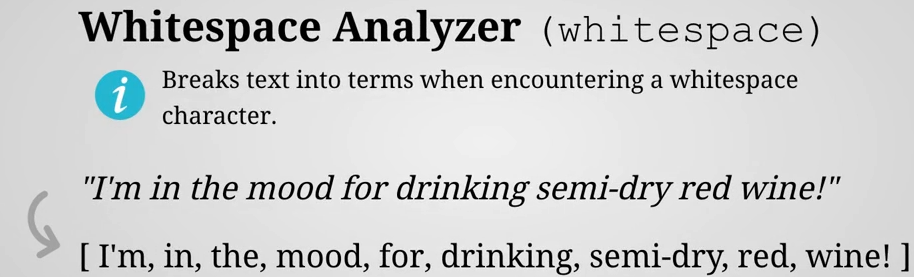
1. ****
2. Now you have seen the character filters, tokenizers and token filters available by default.
3. Let’s take a look at the default analyzers.
4. Analyzer is an orchestration of the parts (character filter, tokenizer & token filters).
5. We already studied **standard analyzer (standard).**



1. It is a wrapper around
   1. The standard tokenizer.
   2. The standard token filter.
   3. Lowercase token filter.
   4. Optionally, the stop token filter.
2. Let’s move on to an analyzer that you haven’t seen before namely the **simple analyzer**.
3. **Simple Analyzer**:  
   
   1. It consists of **lowercase tokenizer**.
   2. This tokenizer is a bit special in that it not only splits input into terms but also lowercase system terms something that will usually be the job of a token filter.
   3. See, it splits the semi-dry into terms when it sees apostrophe which is unlike what you saw with a standard analyzer.
4. **Stop Analyzer (stop):**
   1. 
   2. It consists of
      1. Lowercase tokenizer.
      2. Stop Token filter. See it filtered out “in”, “the” “for”
   3. **NOTE**: This analyzer is quite similar to using a standard analyzer with “**stop token filter**” enabled.
5. Let’s look at a group of analyzers called “**Language Analyzers**”.
6. **Language Analyzers (english, …)**
   1. 
   2. These analyzers are used for analyzing text in specific languages and provide a very easy way of enabling stemming, stop words and more without having to define these parts explicitly.
   3. The example that you see here is for the **English analyzer**.
      1. The standard tokenizer is used.
      2. Stop token filter.
      3. **Stemmer token filter**: So drinking is turned into drink.
      4. **NOTE**: There are other parts to language analyzers but these were the basics of them.
7. **Keyword Analyzer (keyword)**
   1. 
   2. All it does is to take the input and return it as a single term which is done by the keyword tokenizer.
   3. This is useful if you don’t want to tokenize or otherwise manipulate the text fields within documents.
8. **Pattern Analyzer (pattern)**
   1. 
   2. There is also a pattern analyzer which splits text into tokens by matching token separators with a supplied regular expression which is done with the **pattern tokenizer**.
   3. It also lowercases the terms with **lowercase token filter**.
   4. Optionally stop word filter.
   5. In the above example, the regular expression is comma.
9. **Whitespace Analyzer** (whitespace)
   1. 
   2. It is just a wrapper around **whitespace tokenizer**.