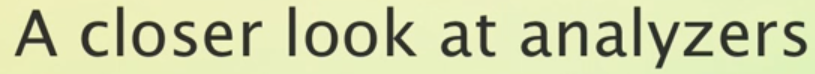
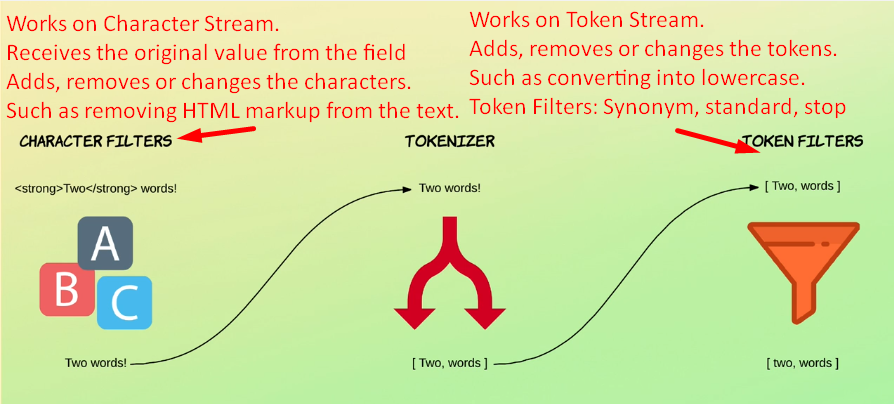
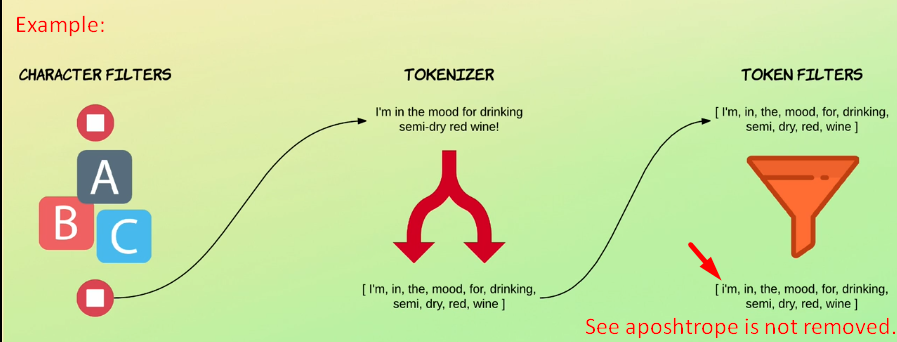
1. 
2. Let’s take a closer look at analyzer.
3. An analyzer consists of 3 things.
   1. Character Filters.
   2. Tokenizer
   3. Token filters.
4. Basically, analyzer is a package of these building blocks with each one of them changing the **input stream**.  
   
5. So, when indexing a document, it goes through the following flow.
   1. **Character Filters**:
      1. First a character filter or more character filters in series are added.
      2. A character filter receives a text field’s original text and then transfers the value by adding or removing or changing characters.
      3. **Example**: To strip out any HTML markup text.
   2. **Tokenizer**:
      1. Then tokenizer splices the text into individual tokens which will usually be words.
      2. So, if we have a sentence with 10 words, we would get an array of 10 **tokens**.
      3. An analyzer can have one tokenizer.
      4. **By default tokenizer named standard** is used which uses a “**Unicode text segmentation algorithm**”.  
         It basically splits by **whitespace** and also removes most symbols such as commas, periods and semi-colon etc.  
         This is because most symbols are not useful when it comes to searching as they are intended for being read by humans.  
         It’s possible to change the tokenizer to preserves the symbols but that’s a rarely a good idea.
      5. The tokenizer is also responsible for recording the following:
         1. Order or position of each term (used for phrase and word proximity queries)
         2. Start and end character offsets of the original word which the term represents (used for highlighting search snippets).
         3. Token type, a classification of each term produced, such as <ALPHANUM>, <HANGUL>, or <NUM>. Simpler analyzers only produce the word token type.
   3. **Token Filters**:
      1. After splitting the text into tokens, it runs through one or more token filters.
      2. A token filter may add (eg. synonyms), remove (stopwords) or change (lowercasing) tokens.
      3. This is kind of similar to a character filter but token filters work with the **token stream** instead of **character stream**.
      4. There are a couple of different token filters with the simplest one being a **lowercase token filter** which just converts all characters to lowercase and other token filters can make use of, such as “**stop token filter**”. It removes common words which are referred to as **stop words**.  
         These are words such as “the”, “a”, “and”, “to” etc. These are the words that don’t really provide any value to a field in terms of search ability because each of these words gives a document very little significance in terms of **relevancy**.
      5. The other token filter is such as “**synonym**” which is useful for giving similar words the same meaning.  
         **For example** the words nice and good have the same semantics although they’re different words. (awful = terrible).  
         So by using the synonym token filter, you could match documents containing the word nice even if you’re searching for the word good. Because the meaning is the same and therefore the document is highly likely to be as relevant even if the query used the other word.
6. All right so far we covered the different parts of analyzers. Let’s take a short moment to walk through an example.  
   
7. When elasticsearch sees a string field in a document, it **configures it as a full text field** and applies the **standard analyzer**.  
   This happens automatically unless you instruct elastic search to do otherwise.  
   The following example is the default behaviour of a standard analyzer with a standard analyzer.  
   **NOTE**: With **standard analyzer**, there is no character filters so the text input goes straight to the tokenizer.   
   The standard analyzer uses the tokenizer named **standard** which does what I mentioned earlier which filters out various symbols and split **by whitespace**.  
   See, this tokenizer splits the input into tokens with special symbols removed. But notice that apostrophe is not (I’m).
8. The array of token is then sent to a chain of token filters. The first one is the token filter named standard. This filter actually doesn’t do anything. Its only purpose is to act as a placeholder in case some default filtering needs to be added in the future.  
   The **stop token filter** for stop words is added by default but it is disabled.  
   This means that the only active token filter that is used with the standard analyzer is the **lowercase filter** which converts all letters to lowercase.  
   **NOTE**: There is an Analysis API which can be used to test the result of applying character filters, tokenizer and token filters. 🡸 Analyzer as a whole.
9. This is very useful for understanding how the various parts of analyzers work.