1. Now, as we have seen a couple of input and output plugins and codes, let’s talk about another **key component** **of Logstash Pipeline -> Filters.**
2. Remember that **filter plugin** performs some sort of processing on events before handing them off to the **configured outputs**.
3. We will not cover all the filters in this lecture because we will introduce filters gradually throughout the course as we need them.   
   So, please consider this lecture an introduction to **Filter Plugins**.
4. One very common **Filter Plugin** to use is one named **mutate**.  
   As the name suggests, it is used to change an event in one way or another, to mutate them.
5. Mutate Filter can perform various actions such as
   1. renaming,
   2. copying fields,
   3. lower casing or upper casing strings,
   4. replace values and more.
6. We will convert a field value from one type to another.  
   Let’s start by converting the quantity field to an integer.
7. Before starting off, let’s see the default behaviour.   
     
   **Sending Event over HTTP to Logstash**:  
   A screenshot of a computer

   Description automatically generated with medium confidence  
     
   **Seeing Processed Output on the Logstash Console Output**:  
   A screen shot of a computer

   Description automatically generated with medium confidence
8. Let’s convert the passed amount to an integer.
9. A screenshot of a computer

   Description automatically generated with medium confidence
10. Restart the Logstash to apply the configuration changes.
11. Let’s send the request again:  
    A screenshot of a computer

    Description automatically generated with medium confidence
12. A picture containing text, screenshot, font, design

    Description automatically generated  
    In case of array, array elements will be converted.  
    Alright, now we know how to instruct logstash to perform conversion.
13. **What is the point of it**?
14. We might want to do this if we’re sending processed events to an output where the data type matters.  
    **An Example** would be sending the processed events to **elastic search** in which case, you want the data types to match any **existing mappings** or for the correct data types to be created dynamically.  
    Perhaps you don’t have the control over the input that is sent to the Logstash.  
     **For example**, we have some legacy system sending data to Logstash and instead of sending integer number, it sends string.  
    You might not have access to modify that system in which you can directly handle with Logstash or perhaps we just want to add a **safeguard** against the wrong data types being sent to logstash in the future.  
    So, this might be the way to protect against the bugs being introduced in those applications.  
      
    That being said, the Logstash is very optimistic in that it expects whatever it receives to actually be some kind of numeric value to the float and integer data types, this means that it will simply cast a supplied value to a number.  
    So, if the type cast is set to “integer” and we pass a string “Hello World”, the result would be zero.  
    If we pass 1003dkjfdkfjd33e3 -> the starting numeric digits will be picked from the beginning and the field’s value will be set to that.  
    In this case, the event processing will continue as normal.  
    This means that convert option should not be used as a way of validating that a field contains a valid data type or not but only for converting one data type to another.
15. We have attached the link to the lecture containing all other options for **Mutate Filter Plugin**.