##### **Groovy goodness**

        So my problem today was this, I had a list of Payment objects, from which I needed to sum the total amount of those which status was "approved", and for better explaining here is my Payment class

public class Payment {

private String paymentId;

private String status;

private BigDecimal amount;

}

        In order to do this on Java I would have to write a loop iterating the array, the for every item ask if the status was approved and then sum the amount in a var, kind of this:

public BigDecimal sumApprovedAmount(){

BigDecimal sum = BigDecimal.ZERO;

for(Payment paument: paymentList){

if(payment.getStatus.equelsIgnoreCase("approved")){

sum += payment.getAmount();

}

}

return sum;

}

        And this is the code using Groovy:

public BigDecimal sumApprovedAmount(){

return payments.findAll{it.status?.equalsIgnoreCase("processed")}?.amount?.sum()

}

        Yes that's right, only one line of code!!!. For those who haven't worked with Groovy I will explain this a little bit how this "GROOVY MAGIC" works.   
The first part is the function findAll that Groovy added to the class List, this method receive as parameter a "clousure", receive a what??, a "clousure" is a term used on Groovy (and most recently in Java 8 as well) and it means that you can pass also a piece of code to a function, instead of just values, and that function will call our piece of code to do his work, in this case the function findAll use our "clousure" to filter the list of elements:

{it.status?.equalsIgnoreCase("processed")}

        Here we specify to the findAll function to take just those elements which status are equals to "processed", and internally Groovy will call this piece of code for every item of the list.  
In the second part we see this:

?.amount?.sum()

        The interrogation signs are for avoid the undesirable NullPointerException, instead of be asking all the time if something is different of null like in Java, using Groovy we just need to use the symbol "?", and the code that follow this will just be applied if this is different of null. And the amount.sum() explains by itself, here we say to Groovy to sum the amount property of all the elements who passed the filter.   
Easy eh??.. that is the "Groovy goodness", see you in the next hack...