Constants

In every business model constants are used. Such constants have a distinct and often commonly recognized name, a justification in law or legislation and a single value that rarely changes. The constant will probably only change because of change in law or legislation.

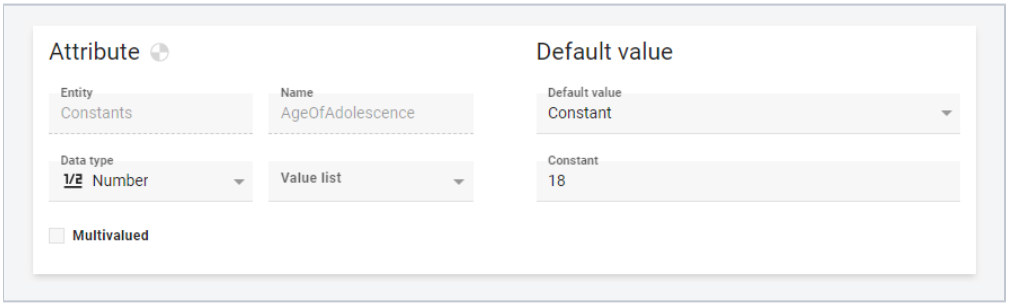
Typical examples of constants are

- The age of adolescence
- The maximum amount of savings
- The percentage of tax due in scale 1
- The percentage of the lower value added tariff
- etc.

There are several ways to organize those constants in a business model.They will be discussed below.

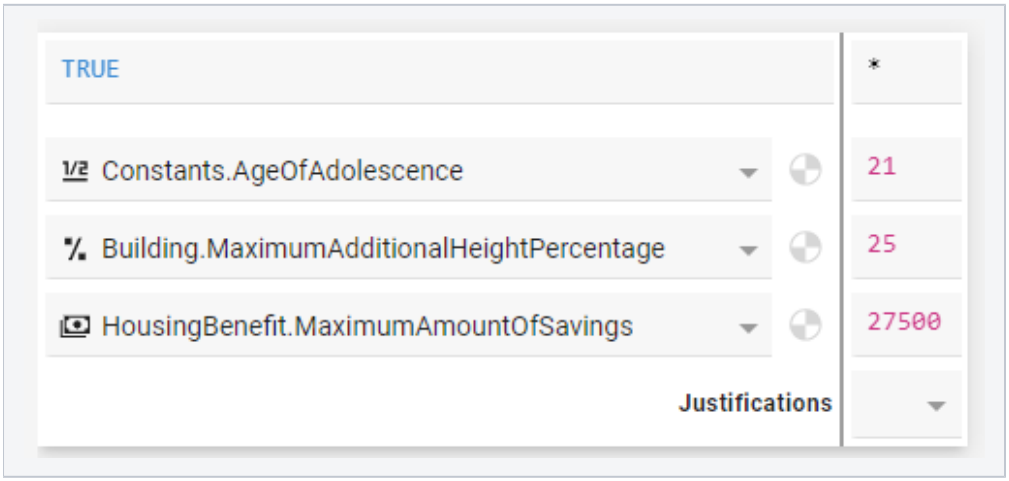
Entity with constants

One way of organizing constant values is combining them in a singleton entity. Of course the attributes of this entity do not share any commonality, except for the fact that they are constants. Business engineers that uphold strict rules of elegant modeling might frown upon such an entity, whereas the more pragmatic business engineers favor one particular space in the business model where all constants are stored and managed.



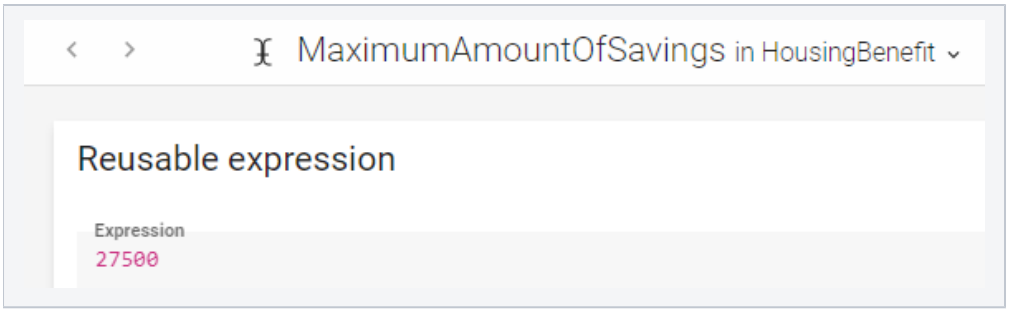
Attributes, set by a single decision table

Another way of organizing constant values is setting them in one single decision table, while the constants themselves are declared wherever the business engineer thinks they belong. This scenario can also be used when using a one entity to store all constants together. The default value of these constants is left out, but one (or perhaps even more) decision table sets the values.



Reusable expressions

Perhaps the most intuitive way of organizing constant values is setting them in reusable expressions. The main reason many business engineers favor this option is that some constants do not belong to an entity. Reusable expressions - used in this manner - are "orphaned" single-valued-attributes; they have the most in common with actual constants.

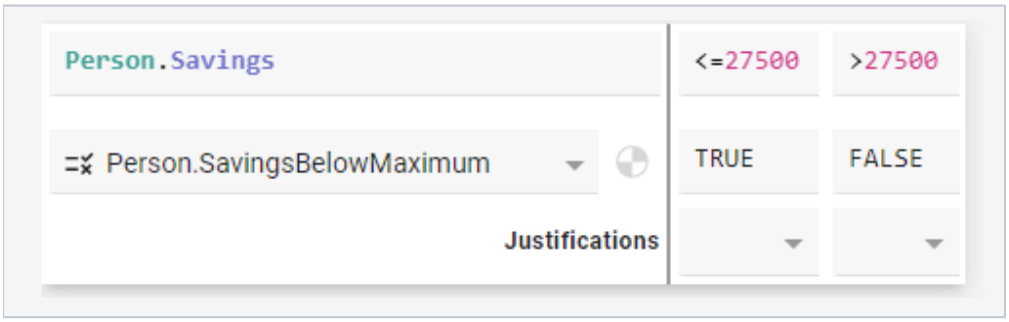


One might argue in favor of using any of the three scenarios discussed here. Other scenarios may be possible as well. There is one con that the three scenarios mentioned here have in common: It is not possible to add a justification for default values and for reusable expressions. Decision tables can have a justification, but only for the complete set of derived attributes (i.e. for one column). If a justification for each constant is necessary, a business rule (or even a decision table) for each constant would work.

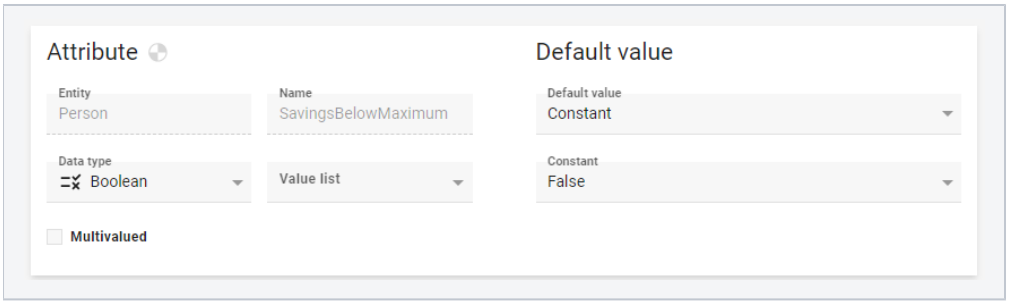
None

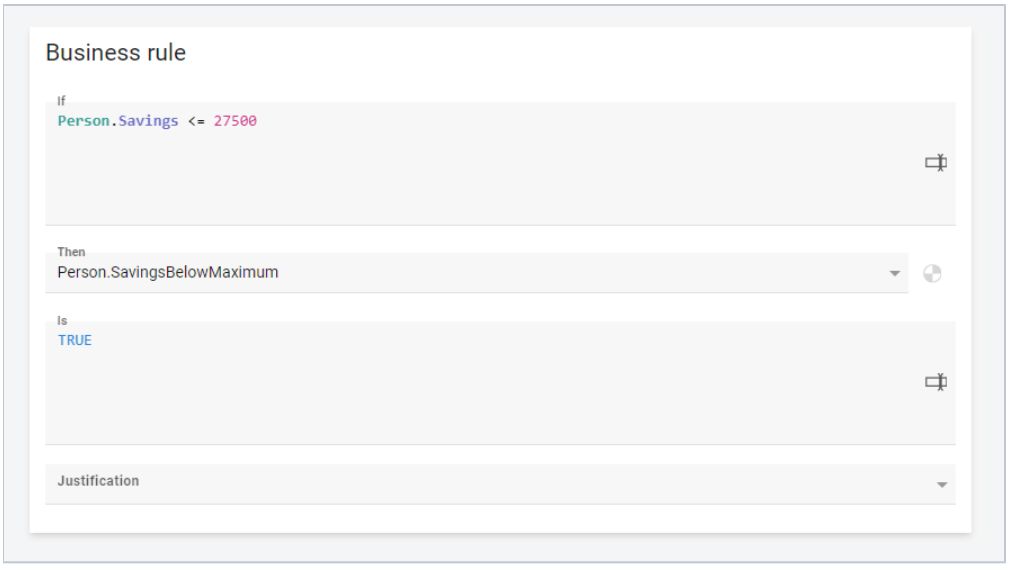
Another possible scenario is to not gather constants at all and just make use of the values that are affiliated with these constants. The con of this approach is clear: when a constant is changed, you will have to update each occurrence of that constant in your model.

See the example below where the maximum amount of savings is not declared as a constant. The value 27500 is merely used in a decision table to determine if a person does not have too much savings.

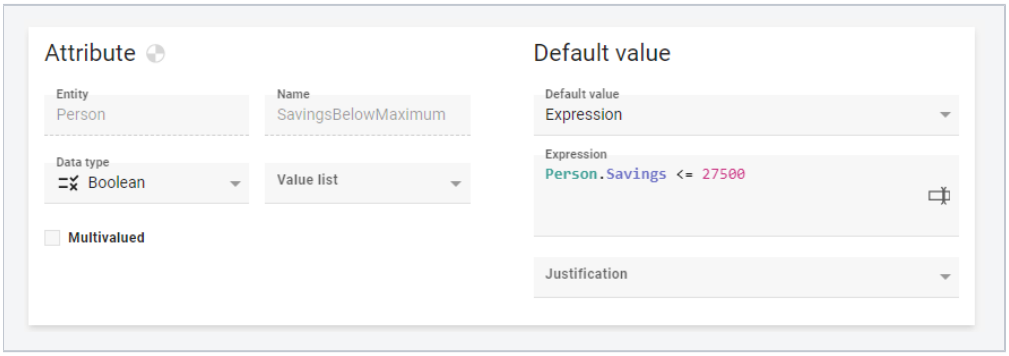


Of course the decision shown above could also have been modeled using a constant default value combined with a business rule or a default value expression, as shown below.





Using a default value expression:



Main chapter: Design considerations

Previous: Dealing with multiple versions of logic

Next: Local variables