**Introduction:**

* Drools is a Business Rule Management System (BRMS) solution. It provides a rule engine which processes facts and produces output as a result of rules and facts processing. Centralization of business logic makes it possible to introduce changes fast and cheap.
* Drools is Rule Engine or a Production Rule System that uses the rule-based approach to implement and Expert System.
* A Production Rule System is Turing complete with a focus on knowledge representation to express propositional and first-order logic in a concise, non-ambiguous and declarative manner.
* A Rule Engine allows you to define “What to do” and not “How to do it.”

**How drools are used in the framework with examples:**

Drools are mainly used for validation purpose and check for the task status in this framework like as shown below:

* To validate the Author details
* To validate the category of the book
* To validate particular view of the book
* To check the status of the task

Examples:

The below example is for Cart, Product and Customer. Based on the business validation the rules will be executed and provides the output.

//Is it out of stock?

rule "Is Out-Of Stock"

when

$cartItem : CartItem(cartStatus != CartStatus.PROCESSED && product.getAvailableQty() == 0)

Then

$cartItem.setQty(0);

End

rule "Verify Qty"

when

$cartItem : CartItem(cartStatus != CartStatus.PROCESSED, qty > product.getAvailableQty())

Then

End

rule "Add Processed CartItem to Order"

when

$cartItem : CartItem(cartStatus == CartStatus.PROCESSED)

then

end

**Advantages of drools:**

* Rules are easier to understand for a business analyst or a new developer than a program written in Java or other imperative-style language.
* Since rules are easier to understand, a developer can spend more time solving the actual problem.
* It is much easier to add new, modify, or remove existing rules than to change, for example, a Java program. The impact this has on other rules is minimal in comparison with an imperative-style implementation.
* It deals better with changes to the requirements or changes to the data model. Changing or rewritingan application is never an easy task. However, thanks to the formalism that rules bring, it is much easier to change rules than to change a Java program.
* The rules are kept in one place (separation of business logic from the rest of the system), which means easier reusability. For example, imagine you've written some validation rules for your application and later on there is a need to do some batch imports of data, so you could simply reuse the validation rules in your batch import application.
* It is possible to change/redeploy rules and processes without even stopping the whole application.

**When not to use a Rule Engine:**

There are a lot of advantages of Rule Engine but that doesn't mean it can be a solution to all the application.

* If your project is small, possibly with less than 20 rules then a rule engine would probably be overkill. However, think twice before making a decision, because many systems start small, but as more requirements are implemented, it suddenly becomes unmanageable.
* If your business logic is well-defined or static and doesn't change often; you don't need to change rules at runtime.