## ## Introduction

What’s a rules engine? It’s a system that executes rules. Okay...so what’s a rule? In its simplest form, it’s an if-then statement that is provided from the outside, rather than being “baked-in” to the engine itself. In the case of OrderCloud.io, we’ll define the “then”s (starting with “order requires approval”) and allow you to define the “if”s via custom **logic expressions**. Effectively, we’re going to allow you to extend platform behavior in much the same way extended properties (xp) allows you to extend the data model.

Here is an example.

Say you want every order over $200 with some specific xp value to require approval from a manager. You would first create a user group containing all approving managers, then create a new approval rule, set the `ApprovingGroupID`, and set the Expression to this:

```order.Total > 200 and order.xp.MyCustomProperty = ‘XYZ’ ```

## ## Supported Operations

A couple things to note:

* `order` supports the same properties as the Order model returned from /orders API endpoints, including xp.
* `=`, `<`, `<=`. `>=`, `<>` comparison operators are supported.
* `and`, `or`, and `not` logical operators are supported.
* +, -, \*, / mathematical operators are supported.
* String values must be enclosed in single quotes.
* Date values must be enclosed in # symbols, i.e. #5/15/2016#
* Parentheses may be used to enclose sub-expressions and control order of execution.

## Line Item Control

What about line items? Glad you asked, because which products are being purchased, in what quantities, charged against which cost centers, etc, are very common in the world of approval rules. But line items are a collection, so we turn to **aggregate functions** to inspect them. Here’s how you would require approval on all orders over $200 charged to cost center ABC:

`order.Total > 200 and items.any(CostCenter = ‘ABC’)`

That’s pretty powerful, but it’s more likely that you only care about the *subtotal* of just the line items matching your CostCenter condition. For this you can use the `items.total` function:

`items.total(CostCenter = ‘ABC’) > 200`

The condition inside the function (called a filter) can be more complex and contain `and`, `or`, etc. just like other parts of the expression:

`items.quantity(ProductID = ‘P1’ or ProductID = ‘P2’) > 5`

It also has access to a special filter that allows you check whether a product is in a certain category:

'items.any(product.incategory("Toys""))'

`items` supports a total of 4 functions:

* `items.any` (true if any item matches filter)
* `items.all` (true if all items match filter)
* `items.quantity` (compare result to a number)
* `items.total` (compare result to a dollar amount)

And one special filter:

* product.incategory('mycustomcategory')

## Complex Approvals

Speaking of functions, there is one defined on `order`:

`order.approved(‘id\_of\_some\_other\_rule’)`

This one’s powerful, because it allows you to set up multi-level approval workflows by chaining rules together. For example, in a larger organization, getting the approval from a department manager might not be enough, and a higher-level VP must also sign off.

All valid elements of rule expressions can be mixed & matched as needed, allowing for very sophisticated rules to be supported:

`(order.Total > 20 and order.approved(‘rule\_id\_1’)) or (not item.any(ProductID = ‘QQQ’) and approved(‘rule\_id\_2’))`

A word of caution: Rules are easy to write and very powerful, but can be very tricky to debug when they don’t work quite like you thought they would. Don’t get more fancy with them than you need to. As always, we’re here to help if you need guidance.

## Where to go from here?

You can also leverage the power of the rules engine to create custom Promotions. Both the promotion EligibleExpression and ValueExpression accept expressions just like the ones described above. For more information, check out this guide(https://documentation.ordercloud.io/use-case-guides/buyer-and-seller-organization-management/create-and-assign-a-promotion).

Now that you have a better understanding of the power of rules, you might be wondering what other areas of the platform might we apply them to? Here are a few ideas currently being kicked around:

* Custom validation (upon creating/editing things)
* Time-based approval rules (aggregated totals over past week/month/quarter, etc.)
* Applying discounts/markups
* Replenish inventory
* Fire off a notification via webhooks

We’d love to get your thoughts on these ideas. Look for significant enhancements to the rules engine and new applications of it in the months ahead.