Pricefx

Visual Configuration

July 2023

Visual Configuration

Basics of Drag & Drop Visual Configuration

Some of our Visual Configuration apps are **powered by the Google Blockly** technology that allows you to configure some aspects of Pricefx by connecting different blocks with drag & drop interface. Although it is very intuitive and easy to use, there might be some hidden features that, if known, would make your life even easier.

This short guide shows how to navigate the **workspace**, use the **toolbox,** drag & drop **blocks** and connect them together in order to create a custom strategy with the Strategy Designer or design an approval workflow with the Workflow Designer.

The screenshots and animations come from the Strategy Designer which leverages the drag & drop workspace extensively, but it works the same everywhere where such a workspace is deployed.

Workspace, Toolbox, and Blocks

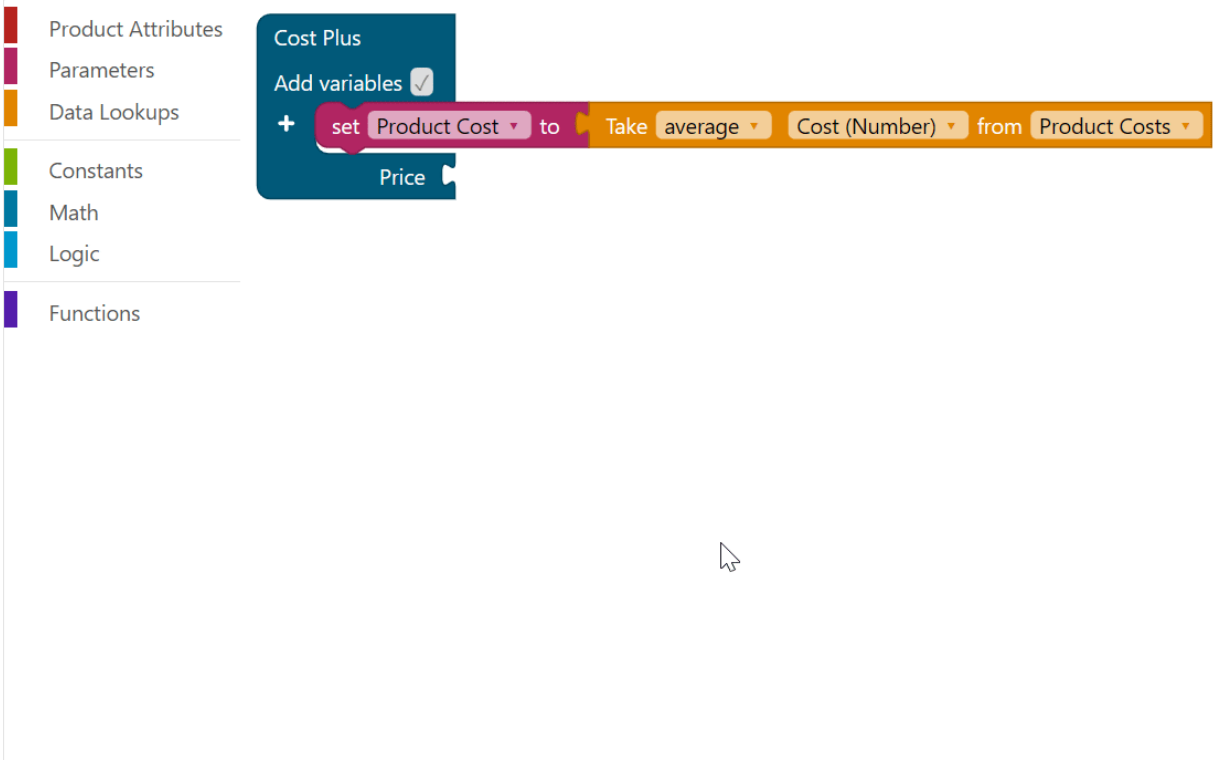
Workspace is an area where you connect the blocks together.

Toolbox is where the available blocks are placed. It is usually located to the left of the workspace, but in some cases, it can be placed on either side or even at the top or bottom.

The toolbox usually contains **Categories** which contain different **Blocks** that can be connected together following certain rules.

Adding Blocks

To place blocks into the workspace, open the toolbox and simply drag them into the workspace.



Deleting Blocks

To delete a block, simply **drag it to the trashcan** in the bottom right corner and release it.

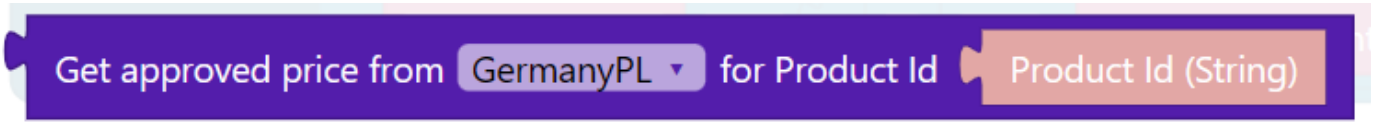
You can also highlight a block and hit the **Delete key** to delete it from the workspace.

Input Types and Block Return Types

Blocks usually cannot be plugged in just anywhere. They can be plugged in with compatible inputs only. Each input has a specified **type** and each block usually has a **return type**. For example, the arithmetic block from the Math category expects only blocks whose return type is a number. You cannot plug in a block that returns a text.

Shadow Blocks

Some blocks in the toolbox have **pre-configured inputs with default values**. For example, the Constrain block in the Math category has shadow numbers and the Get approved price from a price list in the Functions category has a shadow Product ID.



These pre-filled blocks are called shadow blocks. You can tell them apart by their lighter color. They **cannot be removed** from the block, but they can be replaced without needing to remove them first.

In case you want to change a field in a shadow block, you can do it directly. You do not need to replace it with a real block.

Disabled Blocks

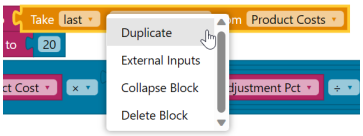
A block can sometimes be disabled. Such a block does not produce any code. You can have these blocks lying around but if you do not plan on using them, it might be better to delete them so they do not clutter your workspace.

Show me

Some Visual Configuration apps will automatically disable blocks that are not connected anywhere; Strategy Designer does this, for instance.

Block Options

Most blocks also have a right-click context menu.



The menu usually contains the following items:

- **Duplicate** – Creates a copy of this block.
- **External/Internal Inputs** – Switches the block to a vertical or horizontal layout. This can sometimes improve readability.
- **Collapse/Expand Block** – Collapses the block showing only its text representation. This can sometimes improve readability.
- **Delete Block** – Deletes the block.

Sometimes, you might also see these items:

- **Disable/Enable Block** – Disables or enables the block.
- **Help** – Opens a website with a contextual help.

Further Reading

You can now dive deeper into the Visual Configuration apps, such as the Strategy Designer Technical Guide or the Workflow Designer TODO.

If you want more information about Google Blockly, see their official website.

Strategy Designer Technical Guide

> This guide is for Strategy Designer version **1.0.0** and newer, released as part of the **Paper Plane** release on June 25th, 2023. **As of July 12th, the latest release is 1.0.1**.

Strategy Designer is a visual interface for configuring selected aspects of the Accelerate Price Setting Package (PSP) which it also depends on.

The main focus of Strategy Designer is to allow business users to **configure custom strategies** that would otherwise need to be coded in Groovy. Strategy Designer provides visual experience powered by Google Blockly technology (open source).

The main features of Strategy Designer are:

Feature	Configures the following part of PSP
Design custom strategies using a drag & drop interface.	Custom Engines
Assign strategies to different product segments and set each strategy's importance.	Strategy Importance
Preview each strategy's result in a Live Preview panel.	PSP does not have this feature.

Deploy the strategies and the importance settings to the partition.	

What Strategy Designer is not:

- **Replacement of the Price Setting module**. You need to manage you data, price lists, and price grids using the standard Pricefx modules.
- **Visual designer for any Groovy logic**. It just allows users to create custom strategies that are handled by the Price Setting Package accelerator. *It does not work independently and cannot be used without it.*
- **Viewer of data stored in your partition**. Although the Live Preview functionality offers you some visibility into your data, it is very limited. Browsing the master data tables or Company Parameters in Pricefx Unity gives you features such as filtering, sorting, storing your preferences, and thus offers much better experience.

Further reading:

- Prerequisites (Strategy Designer)
- Installation (Strategy Designer)
- Design Your Strategies (Strategy Designer)
- Prioritize Strategies (Strategy Designer)
- Deploy to Partition (Strategy Designer)
- Advanced Configuration (Strategy Designer)
- Limitations (Strategy Designer)
- Frequently Asked Questions (Strategy Designer)

Prerequisites (Strategy Designer)

The Strategy Designer aims to be intuitive enough for business users to use, however, there are some concepts you should get familiar with before diving into building your strategies.

These concepts are:

- Knowledge of how Pricefx stores data in Master Data and Company Parameters
- Familiarity with the concept of Price Setting in Pricefx
- Holding business-user knowledge of how Accelerate Price Setting Package works (mainly conceptual knowledge of product segmentation, dependency hierarchies, strategy importance, and pricing strategies)
- Knowledge of Basics of Drag & Drop Visual Configuration

Installation (Strategy Designer)

Strategy Designer is installed automatically together with Price Setting Accelerator using PlatformManager.

A business role StrategyDesigner will be created with the necessary permissions. **Your user account has to be assigned to the StrategyDesigner business role** to see the Strategy Designer menu item under the External Apps menu.

If you want to further adjust the configuration of the installed Strategy Designer, see the configuration options.

If you want to upgrade your version of Strategy Designer, see the instructions.

Design Your Strategies (Strategy Designer)

There are three steps that you need to go through to deploy a custom strategy:

1. **Design** - Design your strategies
2. **Prioritize** - Assign them to product segments and prioritize them
3. **Deploy** - Deploy the strategies and the prioritization to the partition

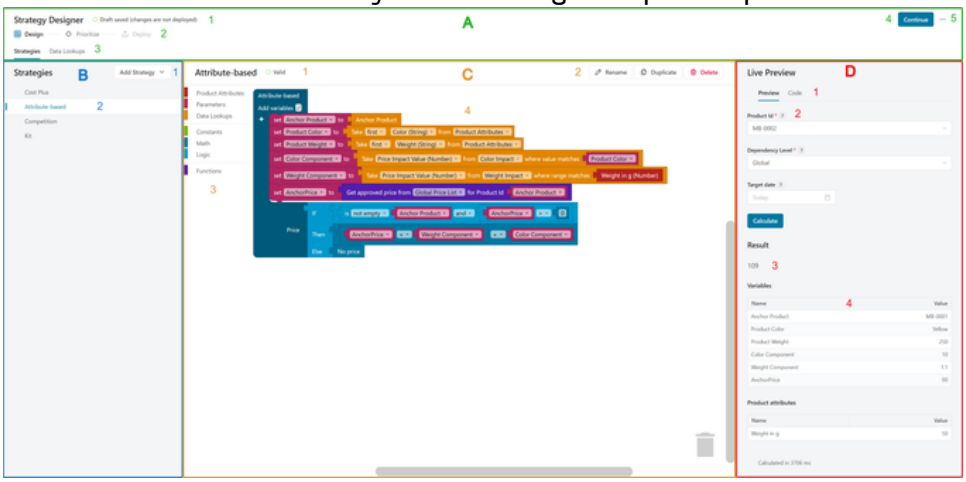
For details see:

- User Interface Overview (Strategy Designer)
- Strategy Templates (Strategy Designer)
- Strategy Variables (Strategy Designer)
- Strategy Toolbox (Strategy Designer)
- Live Preview (Strategy Designer)
- Data Lookups (Strategy Designer)

User Interface Overview (Strategy Designer)

This is the main interface of the Strategy Designer's Design step. It consists of four main sections: Header (present everywhere), List of strategies or data lookups (depending on the selected tab), Workspace, and Live Preview.

If you have not read Basics of Drag & Drop Visual Configuration already, now is the time. The rest of this document assumes familiarity with the drag & drop workspace.



A. Header provides:

1. Status bar informing you about the save/load progress of your configuration
2. Interactive indicator of the current step
3. Tabs where you can switch between Strategies and Data Lookups designer
4. Continue button which takes you to the next step (the same as clicking the steps)
5. Options menu with further actions:
   a. Load the last deployed

configuration from the partition. This throws away your changes and loads the last deployed configuration.

b. Start from scratch. This throws away your changes and cleans your strategies and data lookups.

B. List of strategies or data lookups (based on the selected tab) allows you to:

1. Add a strategy or lookup
2. Select your custom strategies or data lookups from a list

C. Workspace provides:

1. Validation status of the current workspace
2. Buttons to Rename, Duplicate or Delete the current strategy
3. Toolbox containing all the visual blocks you can drag into the workspace
4. Workspace where you put the blocks together

D. Live Preview provides:

1. Tabs - see calculation Preview and Code
2. Inputs for the simulation
3. Calculated result price
4. Calculated intermediate results of the used blocks

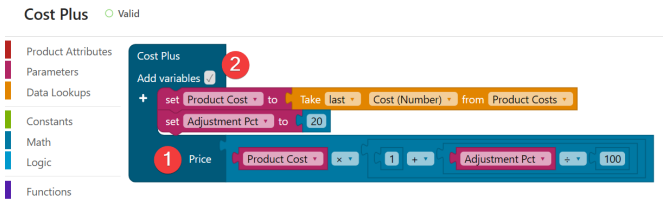
Note: When on a fresh partition, **you start with an empty workspace** – even though you may already have some strategies configured by the Price Setting Accelerator. Strategy Designer allows you to **d efine your custom strategies**, but you cannot configure the PSP out-of-box strategies here.

Strategy Templates (Strategy Designer)

You can create as many custom strategies as you like. You create one by clicking the **Create Strategy** button and either starting with an **Empty** strategy or selecting one of the templates.

The templates are only starting points for your strategies and might give you some head start. They are not fully functioning strategies without further modifications.

This is a very simple Cost-Plus strategy which we configured for demonstration purposes:



Each strategy block has two main inputs:

1. **Price** – Puzzle-like input where you plug your other blocks. Only blocks producing a **n umber** can be plugged here.
2. **Variables** – You can add your Variables to make the strategy more readable and to reuse some of the blocks easily.

The strategy above produces the following formula: *Product Cost x (1 + (Adjustment Pct / 100))* where *Pro duct Cost* is a Data Lookup (more on that later) and *Adjustment Pct* is a Number.

Strategy Variables (Strategy Designer)

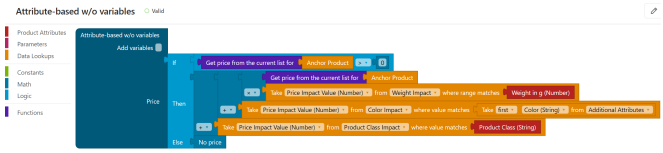
- Purpose
- Adding Variables
- Using Variables

Purpose

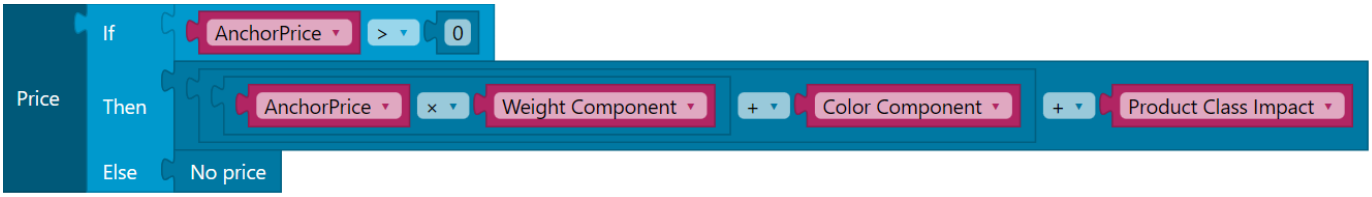
Variables serve two basic purposes:

- Make your strategy **more concise and readable**.
- Make your strategy **more optimized** because a variable is evaluated only once and then its result is reused.

The following comparison shows the same strategy – in the first case built with variables (on the left) and in the second without them (on the right). See for yourself:

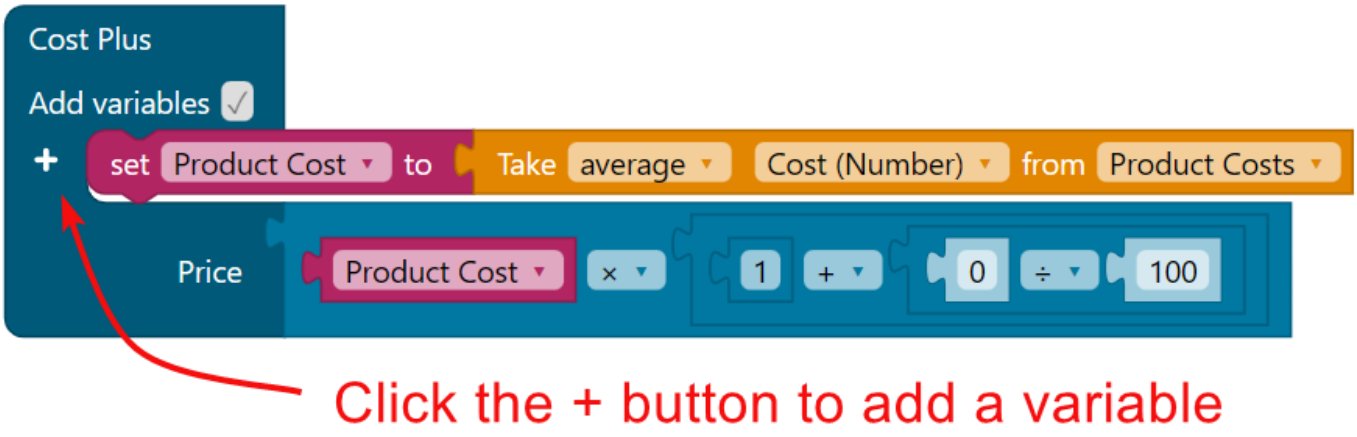


While the strategy with the variables might look "larger" at first glance, the actual calculation plugged into the Price input is much more readable than without the variables.



Adding Variables

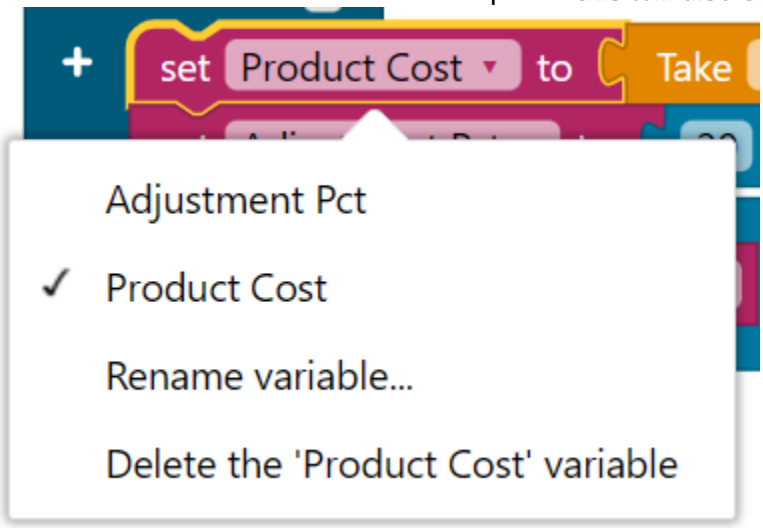
You can add your variables by **clicking the + button** on the left side of the strategy block. A rename dialog will appear where you have to type the variable name. This will add a **variable setter** block into the workspace. You can plug any other block into the variable setter block which will set the variable's value.



> If you have multiple variables and try to drag a setter that has another setter underneath, you will drag both of them. To **drag just a single block**, **hold the Ctrl (Cmd) key before dragging**.

By clicking on the variable setter's drop-down, you will see the following options:

- Change the setter to another variable (note that you cannot have multiple setters for the same variable).
- Rename the variable (a dialog will appear).
- Delete the variable from the workspace – **this will also delete all variable getters!**



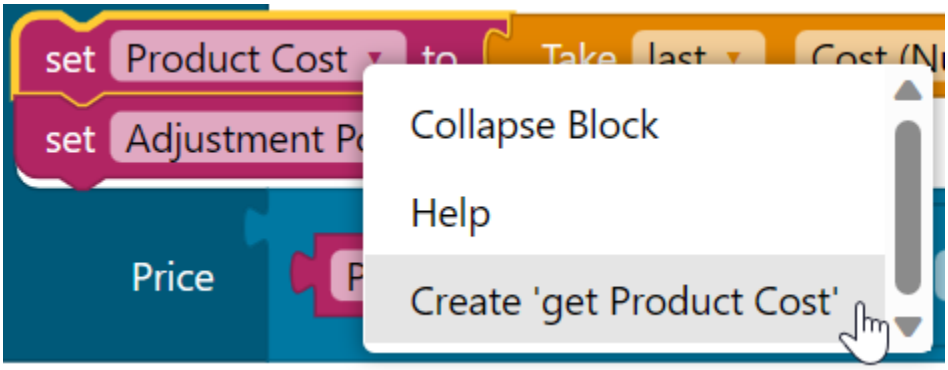
Using Variables

Each variable allows you to use a **variable getter** block which represents the variable's value. You can find all variable getters in the **Parameters** toolbox category and drag & drop them into your workspace.

The variable represented by the getter can be easily changed by clicking the getter's drop-down.

Show me

You can also create a variable getter by **right-clicking the variable setter** and selecting the '*get {Variable}*' option.



Strategy Toolbox (Strategy Designer)

The strategy workspace contains a toolbox with multiple categories. You can drag & drop blocks from these categories into the workspace and plug them into your strategy or other blocks.

Product Attributes

Contains blocks generated from the Product Master table attributes (columns). Each block has its corresponding attribute's return type stated in the parentheses.

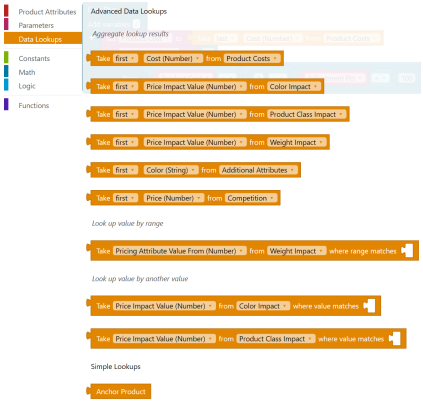
Parameters

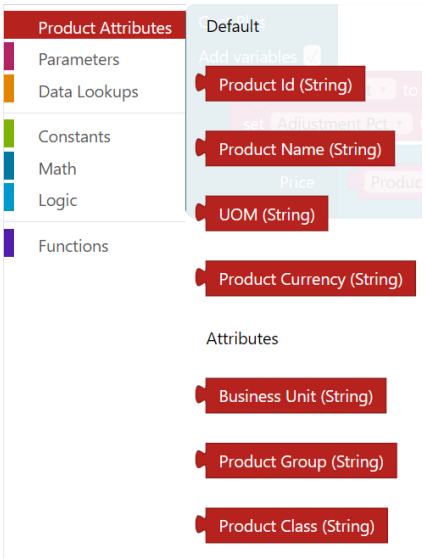
Contains both the variable getters for the defined variables and the out-of-the-box parameters of the Price Setting Accelerator. These always return a number.

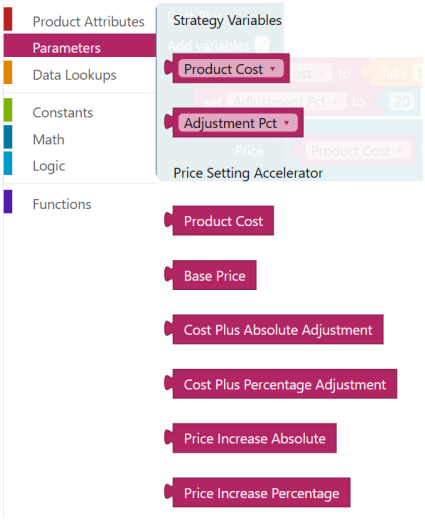
Data Lookups

Contains all of your configured Data Lookups.

There is a section for Advanced Data Lookups (aggregation, range lookups, and value lookups) and Simple Data Lookups.







You need to configure your Data Lookups first in your Data Lookups tab to see anything here.

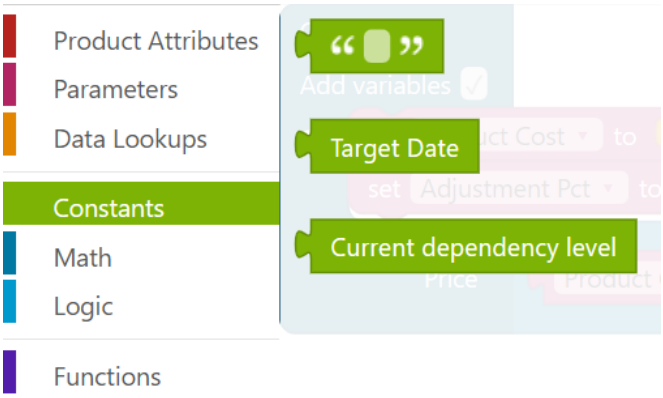
The PSP parameters must be configured properly in PSP before they can be used in a strategy. If not configured, the strategy will fail with errors.

You can hide some or all the PSP parameters in the configuration options.

Constants

Contains the following constants:

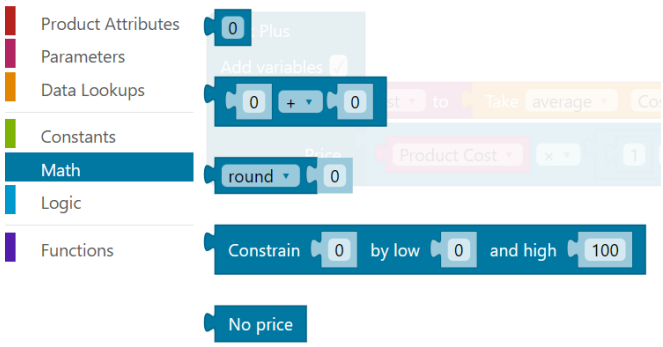
- **Text** block
- **Target Date** block which represents the target date of the calculation when the strategy is used in a PL or LPG or the Live Preview panel
- **Current dependency level** block represents the current dependency level of the PL or LPG being calculated



Math

Contains blocks for mathematical operations, and all accept numerical inputs.

- **Number** block
- **Arithmetic** block
- **Rounding** block
  (round = half-up, round up, round down)
- **Constrain** block – Limits the first value by a minimum (low) and maximum (high), both optional.
- **No price** block – Use this block as the final price of the strategy in case you do not want it to return any price. Such a strategy will be skipped by the PSP.



Logic

Contains blocks related to logical operations.

- **If-Then-Else** block - If the value in the If input is true, then return the value in the Then input, otherwise return the value in the Else input.
- **Evaluate** block (aka Switch-case block) allows you to evaluate a value plugged into the first input and then specify multiple If conditions to match the value being evaluated. The return value of this block is the return value of the first matched If block.
- **Logical and/or** block used to group multiple conditions together.
- **Comparison** block used to compare two values.
- **Is (not) empty** block returns true if the given input is (not) empty (null or empty text).
- **Boolean** block returns either true or false.

Functions

Contains miscellaneous functions and custom Groovy functions.

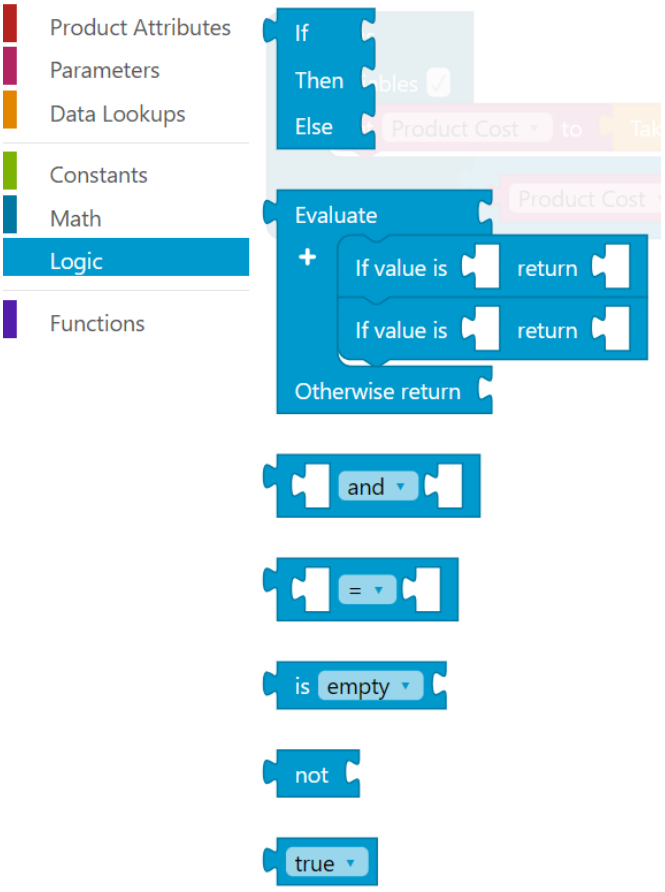
Price List/Grid functions

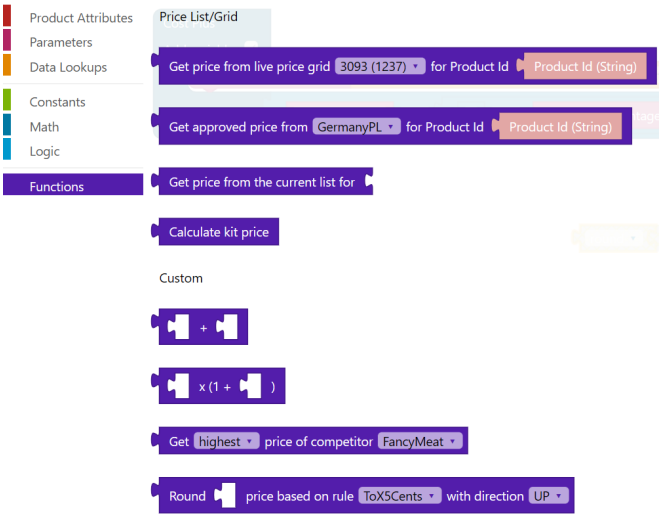
- **Get price from live price grid** returns a price for the given Product ID from the specified Live Price Grid or zero if the price is not there.
- **Get price from approved price list** returns a price for the given Product ID from the specified approved Price List. Only approved price lists are available.
- **Get price from the current list** returns a price for the given Product ID from the Price List /Grid currently being calculated. This triggers a second-pass calculation. Usually used to calculate a price of an anchor product in attribute-based pricing.
- **Calculate kit price** returns a kit price based on the current product's BoM list. All of the kit parts must be in the same Price List/Grid which is being calculated. It triggers a second-pass calculation.

Custom functions

These blocks are generated from a custom Groovy library containing block definitions and code. They appear only if they are configured.

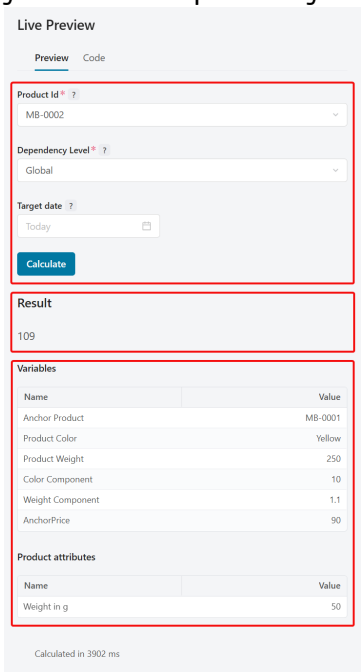
Here is the documentation on how to configure your custom functions.





Live Preview (Strategy Designer)

The Live Preview function allows you to **see the calculated results of your strategy or data lookup in real-time**. It also provides useful **validation messages** in case your workspace is not properly configured. The panel with the Live Preview is located on the right side of the user interface. By default, it is hidden, and you have to expand it by clicking on the arrow button.



You can switch between the **Preview** and **Code** tabs:

- **Preview** – Allows you to perform a simulation of the current strategy.
- **Code** – Shows the code generated by the strategy. This tab is for development and debugging purposes only. Business users should not pay much attention to it.

The Preview panel is divided into three main parts:

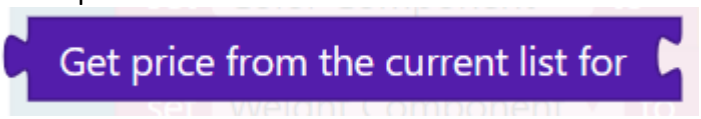
Inputs

This section contains the inputs required to run the simulation. Some inputs are generated only if the strategy requires them.

- **Product ID** – Select a product to simulate the strategy with. Required input, although some strategies and data lookups might not need it.
- **Dependency Level** – Select the dependency level to simulate the strategy with. **Only independent levels are supported** at the moment.
- **Target Date** – Select a target date to simulate the strategy with. If not specified, today's date will be used.

The following inputs are only generated if certain blocks are present in the workspace:

- **Simulated price from the current list** – Required if the *Get price from the current list* block is in the workspace. Because this block triggers a second-pass calculation, it cannot be simulated easily. The value from this input will be used for the simulation of this block's value.



- **Simulated kit price** – Required if the *Calculate kit price* block is in the workspace. Calculating the kit price also requires a second-pass calculation so we simulate it instead in the live preview.

To calculate the strategy, click the **Calculate** button. The button will be enabled only if all the required inputs are filled. Once filled, the strategy will be automatically recalculated when the workspace changes.

Result

Shows the **calculated result price** of the strategy. The price is always rounded to two decimal digits. Hover your mouse over the price to see the non-rounded price.

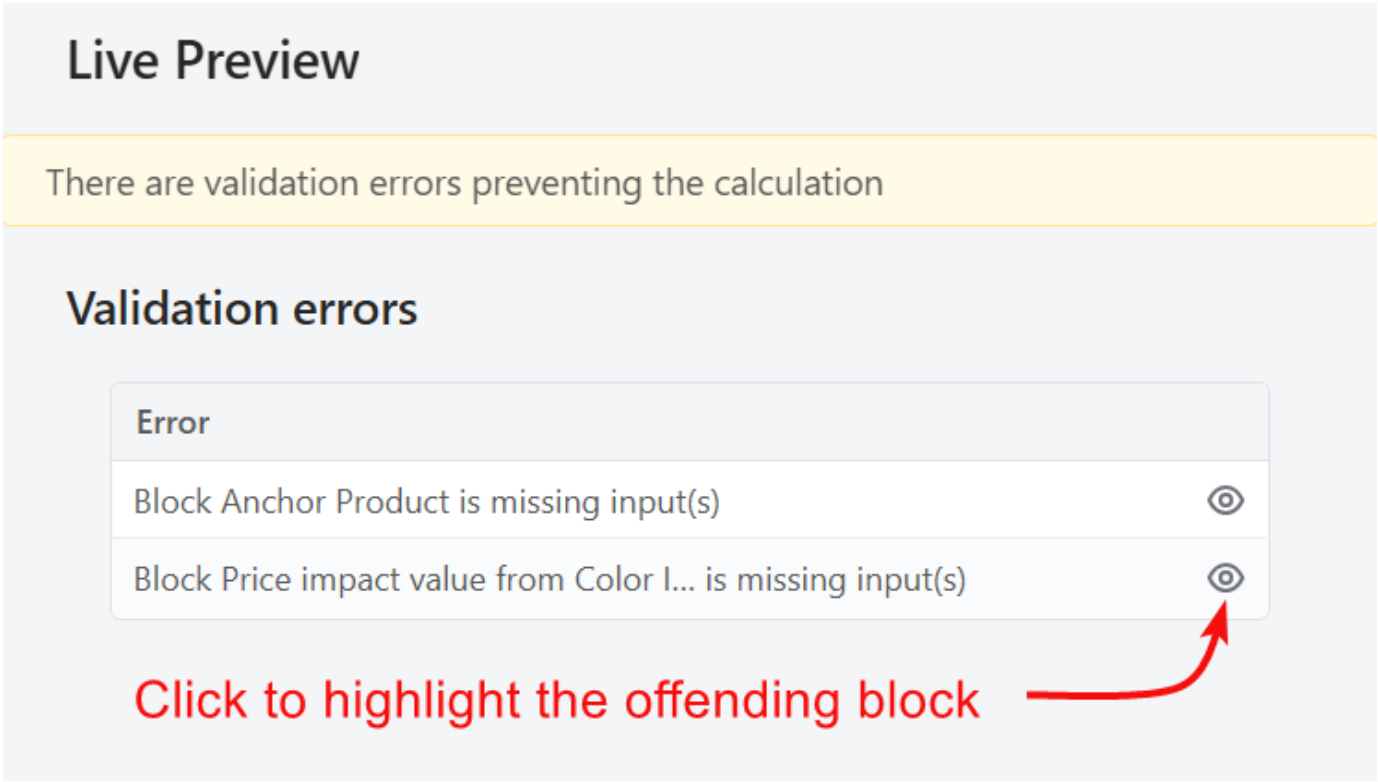
Intermediate Results

Shows multiple tables with calculated results of each block in the workspace, such as variables, product attributes, PSP attributes, and data lookups. This gives you the possibility to debug your strategy in real-time.

Validation

If the workspace is not properly configured, the Live Preview panel will show **validation errors** instead of results. In such a case, you will not be able to see the simulated price or the code until you fix the errors.

You can **click the eye icon to highlight the offending block** in the workspace.



Data Lookups (Strategy Designer)

Data Lookups are a crucial concept in Strategy Designer. They allow you to **specify a data set that can be reused across your strategies in different ways**. There are two kinds of lookups: **simple** and **advanced**.

A general guide to deciding whether to use the simple or advanced look-up:

- Use **simple lookup** if you only need to **fetch a single row for the SKU currently being calculated**, and pick a value from it **without any transformation**.
- Use **advanced lookup** if:
  - you want to **fetch data for a different SKU** than the one being calculated,
  - you need to **filter, sort, or aggregate** the data,
  - you need to **look up a value based on a range** or a value which comes from another data lookup.

You can create a Data Lookup by clicking the **Add Lookup** button above the list of lookups.

In this section:

- Data Lookup Status

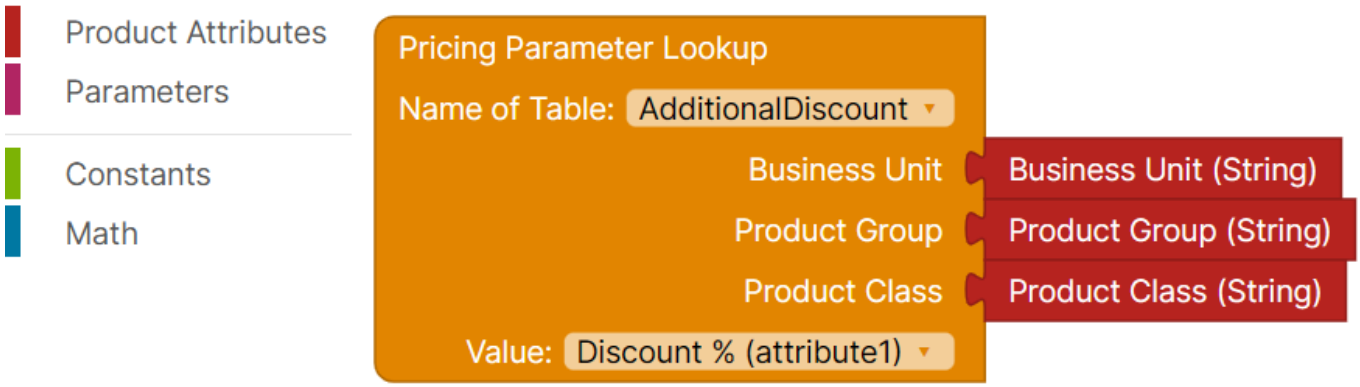
Simple Data Lookups

Product Extension & Company Parameter Lookup

These two simple lookups can be compared to Excel's VLOOKUP function, simply fetching a single row based on a key or a set of keys. They always require all key fields to be specified and always produce just one value. They do not allow filtering, sorting, or aggregating data.

The Product Extension and Company Parameter lookups work almost the same, with just one slight difference: the **Product Extension lookup** is always **filtered by the SKU of the currently calculated product**.

Below is an example of a simple lookup of a company parameter table called 'AdditionalDiscount'. This table identifies each value based on three keys: Business Unit, Product Group, and Product Class. There is an input for each key that needs to be specified. Based on the specified set of the three keys, the lookup identifies a single row and returns its column called 'Discount %'.



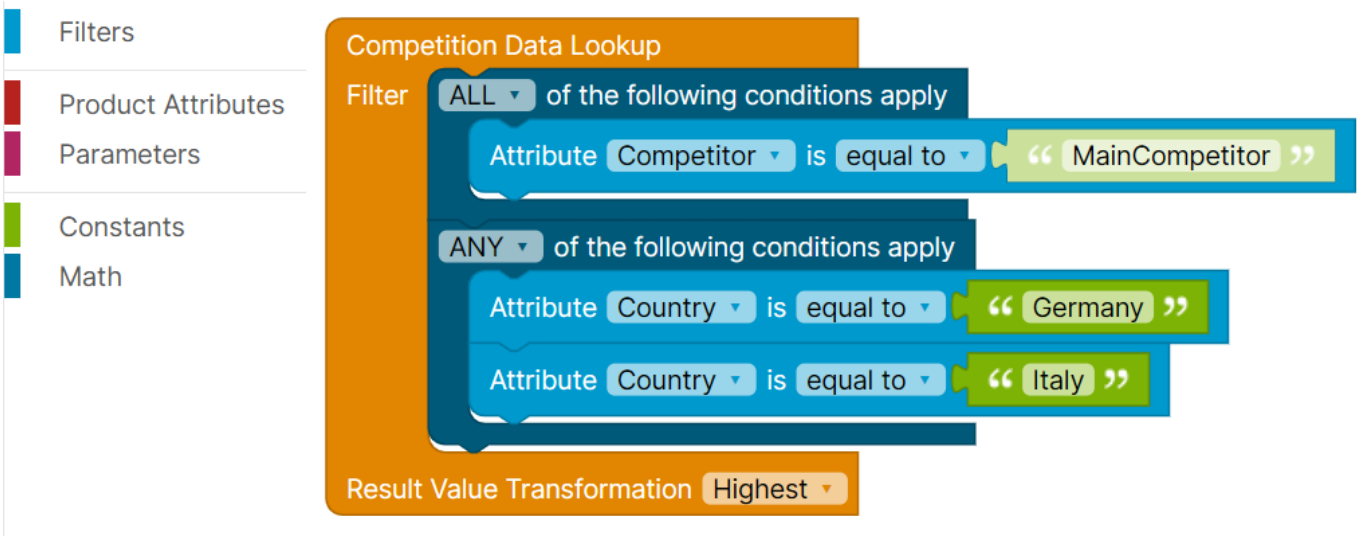
If you need to perform a Product Extension lookup for a different SKU than the one currently being calculated, or if you need to filter, sort, and aggregate your data, you need to use the advanced data lookup.

Competition Lookup

The Competition Lookup looks into the Competition Data table and allows you to filter records based on its columns. It is **always filtered for the SKU currently being calculated** and always returns a price, either the highest, lowest, or an average of all the records that the lookup produces.

Notice the Filters category in the toolbox that allows you to drag & drop the filter criteria. Everything within the logical block with the 'ALL' option selected will be logically AND-ed together. Everything within the logical block with the 'ANY' option will be OR-ed together. Everything which is directly plugged into the Filter input is AND-ed together.

In the example below, the filter could be rewritten like this: Competitor = 'MainCompetitor' AND (Country = 'Germany' OR Country = 'Italy).



If you need more filtering, sorting, or aggregation options, you need to use the advanced data lookup.

Advanced Data Lookup

The advanced data lookup works in a different way compared to the simple lookup. It **splits the data fetching from the aggregation**.

- The data fetching part is configured in the Data Lookups tab. An advanced data lookup **does not produce a single value but always fetches a set of rows**. You can **filter and sort** the result as you wish.
- The aggregation part happens in your strategy. Here you **determine how to transform the rows to produce a single value**.

The flexibility, however, comes with a price, and that is that the advanced data lookup is generally slower.

> **Do not use the advanced data lookup if you can achieve the same result with the simple one.** The advanced data lookup is generally slower than the simple one.

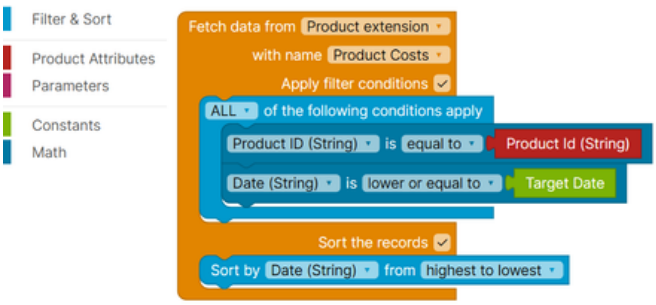
The advanced data lookup supports these data sources:

- Company Parameter
- Product Extension
- Product Master
- Customer Extension
- Customer Master
- Competition Data

Below is an example of the advanced data lookup. The source is a Product Extension called 'Product Costs'. The following filter conditions are applied:

- SKU column is equal to the ID of the product currently being calculated,
  and
- Date is lower or equal to the target date,
  and
- Dependency Level Name is 'Global'.

Additionally, the result is sorted by the Date from the highest (most recent) to the lowest (oldest).



Result

Product ID	Cost	Date	Dependency Level Name
MB-0001	€12.00	2019-01-01	Global
MB-0001	€10.00	2019-01-01	Germany
MB-0001	€15.00	2019-01-01	France

This lookup can return zero, one, or multiple rows, as we can see in the Live Preview, and it returns all columns that are present in the table.

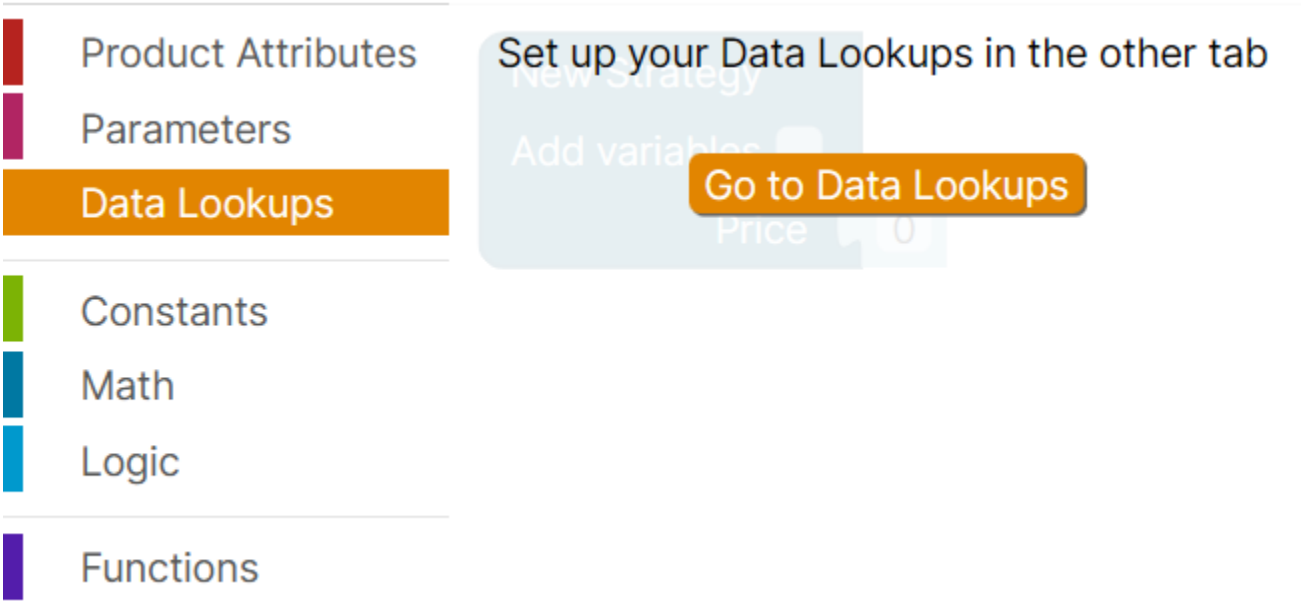
Because of this, **we can define such a lookup once and use it multiple times in our strategies**, each time querying a different column and a different aggregation function.

Using Data Lookups in Strategy

General Information

Before you can use your data lookup in a strategy, **you have to save it**. This is done by clicking the Save button. **Even when you rename a lookup, you need to save it** for the new name to be reflected in the strategies. After you save a lookup, a **corresponding block will appear in your Strategies workspace in the toolbox** under the Data Lookups category.

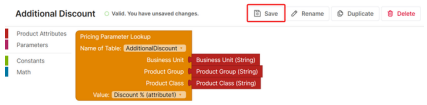
If you do not have any saved data lookup, the Data Lookups category will be empty.

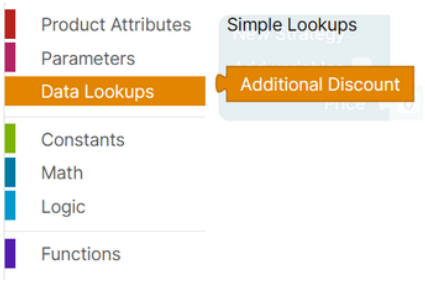


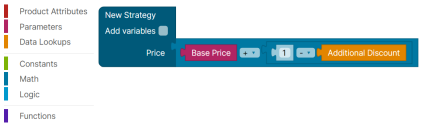
Using Simple Lookups

After you save a simple data lookup, it will be available in the Data Lookups category in the strategies toolbox as a block with no configuration options. This block returns the value specified in the data lookup' s configuration.

Below is an example of a simple data lookup into the company parameter called 'AdditionalDiscount'. This data lookup returns a 'Discount %'. When you use the Additional Discount block in your strategy, it will evaluate the lookup based on the currently calculated product and return the found discount. You can then use it in your strategy.



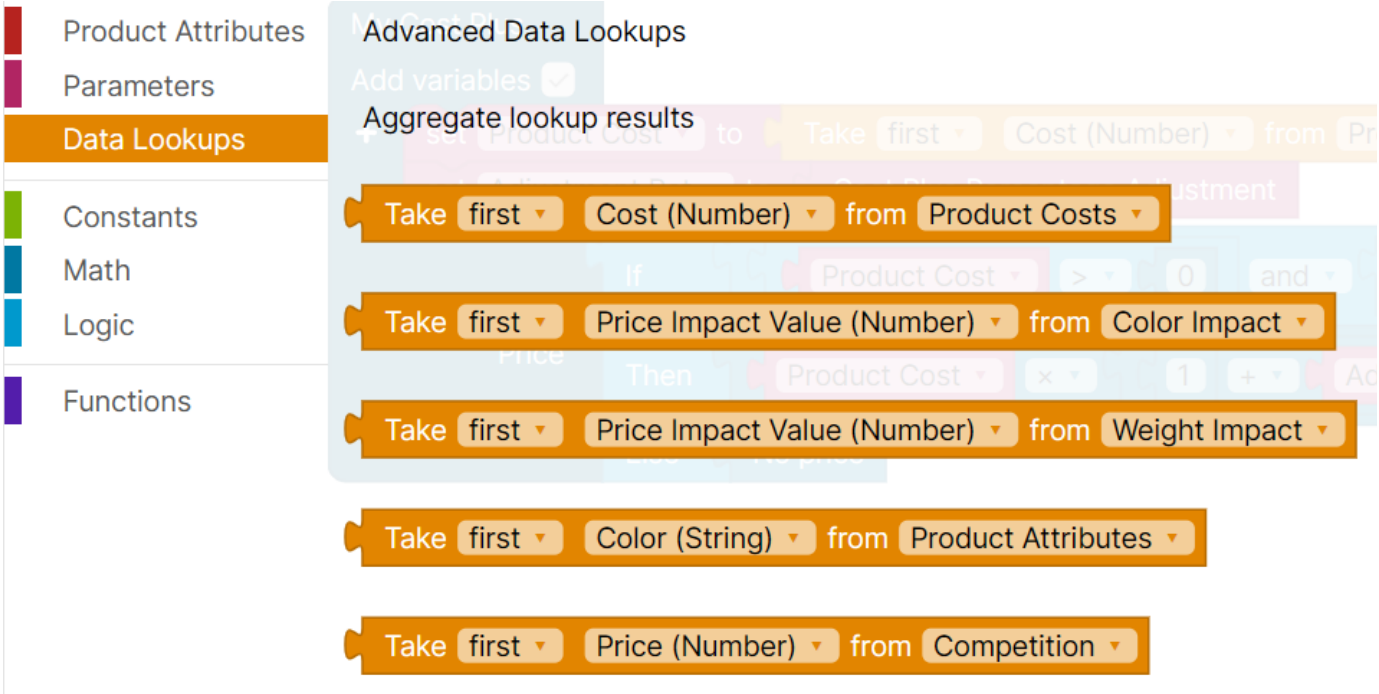




Aggregating Values from Advanced Lookup

As we described earlier, the advanced data lookup does not return a single value but a set of rows with multiple columns. When it is being defined, it does not know yet what column will be used and how the values of that column will be aggregated. This is decided when the data lookup is used in a strategy. Therefore, **each saved advanced data lookup creates an aggregation block** that allows us to select which data lookup we want to use, which column we are going to aggregate, and which aggregation function we use.

Below is an example of several aggregation blocks, each produced for a saved advanced data lookup.



Remember the Product Costs lookup we specified earlier, which returned multiple rows? Here we can decide how to use it. For example, we can take an average of the values in the Cost column.

There are many aggregation functions available. You can reuse the same advanced data lookup multiple times, each with a different column and aggregation function.

Range Lookup

Sometimes you might want to filter a lookup based on a value that is not yet known at the time when the lookup is defined.

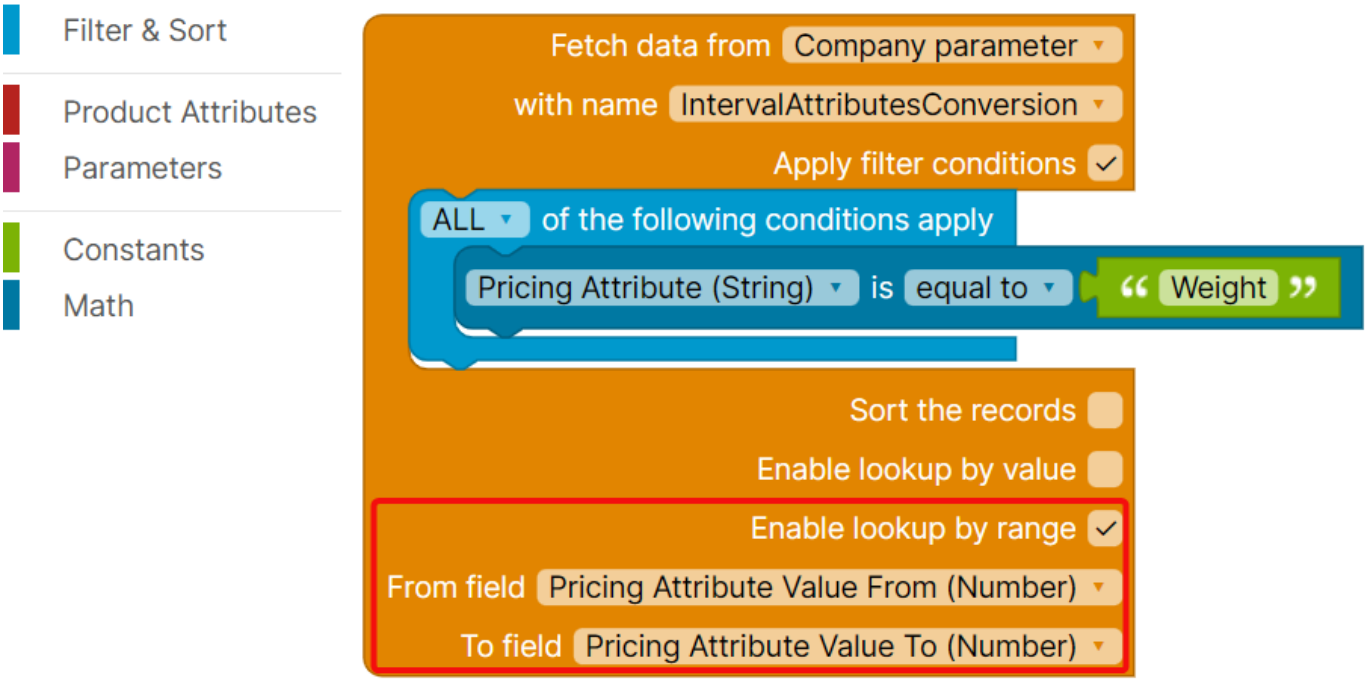
Example: We want to calculate a price impact based on a certain product's weight. We store a price impact value for multiple weight ranges. Therefore we define a lookup that produces these three rows:

Result

Pricing Attribute	Pricing Attribute Value From	Pricing Attribute Value To	Price Impact Value
Weight	0	200	1.1
Weight	200	300	1.2
Weight	300	999999	1.5

However, the problem is that product weight comes from another lookup and therefore cannot be directly used in this lookup's filter.

In such a case, we can **enable lookup by range** for this data lookup by checking the corresponding checkbox. When we do that, we are required to specify which columns define the range. In our case it is the columns 'Pricing Attribute Value From' and 'Pricing Attribute Value To'. By doing this, we made the advanced data lookup a little smarter because now it knows how to perform a range lookup.



This opens a new possibility and creates a new block type in the Data Lookups toolbox in the strategies. This block has an additional input where we can specify the value used for the range lookup. In our example, it would be the weight of the product.

Look up value by range

Let's say we have a Product Weight coming from a different lookup, stored in a variable. In our range lookup, we plug this variable into the block so the lookup knows which value we are comparing the range to. Do not forget to take the correct result column, which is not the 'Pricing Attribute Value From' but the 'Price Impact Value'.

As you can see, the Product Weight was 120, therefore the range lookup returned a Price Impact Value of 1.1. When multiplied by the Product Cost of 12, the resulting price is 13.2.

⚠ If multiple records fit the looked-up range, the block might select one of them at random. Generally, when this happens, it suggests that your data is not defined very well because the ranges overlap and you should fix the data.

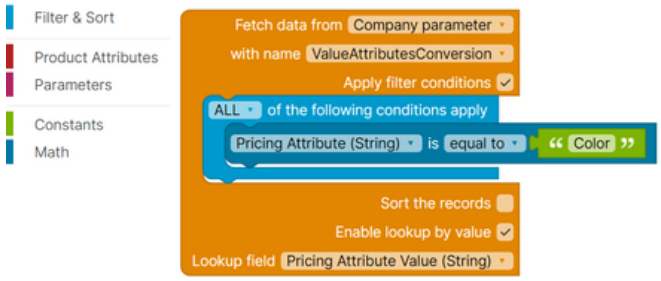
Value Lookup

Sometimes you might want to perform a lookup not by a range, but by a single value. The setup is very similar to the range lookup, only this time you check the **Enable lookup by value** checkbox. When you do that, you need to specify the **Lookup field** which will be used for such a lookup. The Lookup field **should contain unique values** only.

In the example below, we have a table with a Color attribute with different colors stored in the Pricing Attribute Value column. Each color has an impact stored in the Price Impact Value column. We need to tell the data lookup that the column we want to use for the value lookup is the Pricing Attribute Value column.

Result

Pricing Attribute	Pricing Attribute Value	Price Impact Value
Color	Green	12
Color	Red	5
Color	Yellow	10



When we do this and save the lookup, a new kind of data lookup block will appear in the Data Lookups category in the strategies toolbox. It works in the same fashion as the range lookup, only this time the record with the matching 'Pricing Attribute Value' will be picked.

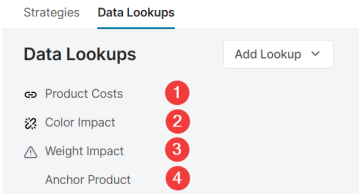
Look up value by another value

If multiple records fit the looked-up value, a random one might be picked as the result. Generally, when this happens, you get a suggestion that your data is not defined very well and you should fix it.

One advanced data lookup might be enabled for both the range lookup and the value lookup. Just make sure the ranges do not overlap and the values in the Lookup field are unique.

Data Lookup Status

When you start using the strategies, you will notice some icons in the list of data lookups. Here is their meaning:



1. **Link** - This data lookup is used in one or more strategies. Hover over this icon to see how many times this strategy is used.
2. **Broken link** - This data lookup is used in one or more strategies, but has unsaved changes. You can either save them or revert to the previously saved version.
3. **Triangle with an exclamation mark** - This data lookup has validation errors. The icon is shown regardless of whether it is saved or not.
4. **No icon** - This data lookup is not used anywhere yet.

Data lookups which are used in one or more strategies **cannot be deleted**. Also, you cannot change some properties of these lookups, such as whether they are enabled for a range or value lookup.

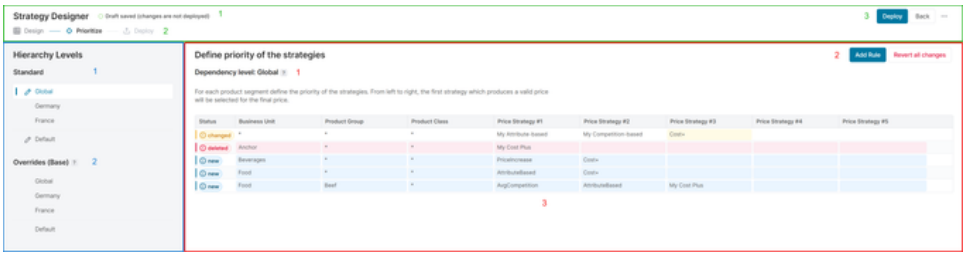
Prioritize Strategies (Strategy Designer)

Design — Prioritize — Deploy

When you are finished designing your strategies, you can proceed to the next step which we call Prioritize. This is where you assign strategies to each product segment and prioritize them. It is good to have an understanding of how Strategy Importance and Dependent Price Lists and Data Fallbacks work.

This step is only accessible if all of the strategies are valid. Otherwise the Continue button will be disabled.

User Interface Overview



Header (same as in the Design step) provides:

1. Status bar informing you about the save/load progress of your configuration
2. Interactive indicator of the current step
3. Deploy button which deploys the current configuration to the partition

List of Hierarchy Levels shows:

1. Standard Levels
2. Override (Base) Levels

Strategy Prioritization Table shows:

1. Information about the selected dependency level
2. Table control buttons

- **Add Rule** button which adds new rows to the table
- **Revert All Changes** button which reverts all changes made to the selected level

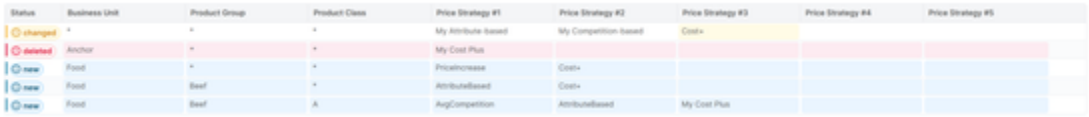
3. Table with product segments and prioritization

Strategy Prioritization Table

The biggest part of the screen belongs to the strategy prioritization (or importance) table. This table contains **rules that determine which strategies are calculated for which product and in what order**.

How Prioritization Works

Each product falls into one of the product segments, which are defined by the Price Setting Accelerator. In the example below, there are three product segments: Business Unit, Product Group, and Product Class. Each product has these attributes and therefore falls always within one segment.



In order to determine which rule applies, the product's attributes are compared with the segment, and **the rule with the most specific matching segment is chosen**. For example, if we have a product with Business Unit = Food, Product Group = Beef, and Product Class = A, the last rule will be picked. If the product has Product Class = B, then the second from the bottom rule will be picked, and so on. If none of the rules applies, the topmost rule is picked as it covers all segments.

When a rule is determined, the Price Setting Accelerator calculates all the strategies in the left-to-right order. The **final price of the calculation will be the price of the first strategy that returns a valid price**.

In our example, say the product does not have a competition and the AvgCompetition strategy does not return a price, but the AttributeBased strategy and the My Cost Plus strategy do return one. In such a case, the final price will be the price of the AttributeBased strategy.

Controlling the Table

When editing the table, the following restrictions apply:

- The product segment columns will only allow you to pick a combination that does not exist in the table. Therefore, **you cannot enter duplicate data**.
- You cannot have a wildcard to the left of a specific value. **Wildcards can be placed only to the right of the more specific values**. This is the current limitation of the PSP.

In the table, you can:

- **Add a new rule** by clicking the Add Rule button. This will add a new rule at the bottom of the table and pre-fill it with the next unused segment.
- **Change a rule** by picking another value from the drop-down in the cell you want to change. The restrictions above apply.

- **Delete a rule**. To do so, hover over the row which you want to delete and click the Trash icon on the right side. Such a row will be disabled and further changes will be prevented.
- **Revert all changes** by clicking the Revert all changes button above the table. This will revert all changes you made to this table so far.

Hierarchy Levels

The **items in the list of hierarchy levels are fixed and depend on how many levels you have configured in the PSP**. You might have only the Default level configured, or you can have more levels. For details on how to configure this, see Dependent Price Lists and Data Fallbacks. Generally, this is configured when the PSP is deployed for the first time on the partition.

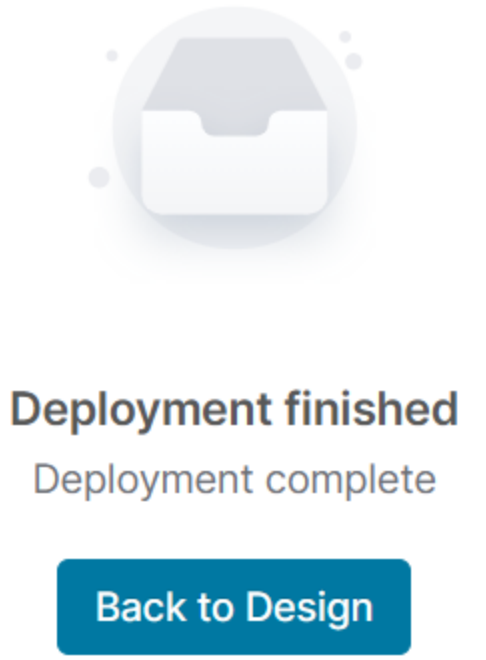
In short, you can have different settings per dependency level (or context) in which your price lists and grids are calculated. For example, you can have different settings on a Global level and Country level.

Standard vs. Override (Base) Levels

Rules defined in an Override table always take precedence over rules defined in the Standard table of the same level. Usually, the Standard tables contain rules that are not changed very frequently, and the Override tables contain rules which change a lot. An example of rules in the Override table would be a temporary promotion. Instead of rewriting your Standard Table, you can just add a rule into the Override table of the same level. If two rules with the same segment are found in both the Standard and the Override tables, the rule from the Override table takes precedence. Only if it does not return a price, the rule from the Standard table is evaluated.

Deploy to Partition (Strategy Designer)

Once you are satisfied with the strategies and their prioritization, you can deploy everything to the partition by **clicking the Deploy button**. A confirmation dialog will appear and after confirming, you will be taken to the Deploy screen which will give you information about the deployment progress.



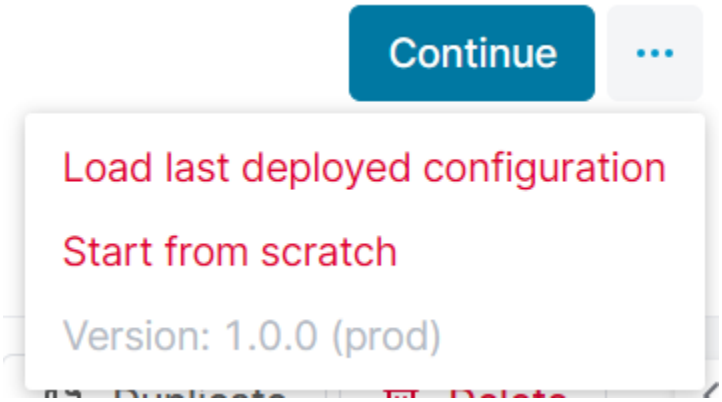
If no error was encountered, you will be presented with the success message and a button that will take you back to the first step while reloading all the data from the freshly updated partition.

If an error was encountered, it means the deployment was unsuccessful. This can happen mainly because of a wrong configuration of the Price Setting Accelerator. In case you encounter an error, write down the error message and send it to the support team.

Draft and Deployed Configuration

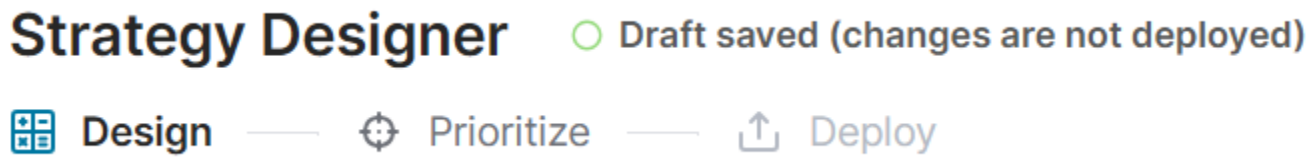
There are two kinds of configuration stored on the partition, the **deployed** one and the **draft**. When you open Strategy Designer, first it will look for a draft version available on the partition. If there is one, it will be loaded. If not, the last deployed configuration (if present) will be loaded.

You can always **discard the draft and load the last deployed configuration** from the partition. This is done from the three-dot menu in the top-right corner.



Each time you make a change to the configuration (list of strategies, lookups, blocks), it is **automatically saved to the partition as a draft**. There is one draft per user, which means that **each user can have their**

own work-in-progress version of the strategies. You can see the status of your configuration in the status bar in the app's header.



There is, however, **just one deployed configuration** stored on the partition. This means that whenever a user deploys their draft to the partition, it **will overwrite the previous version** of the deployed configuration. And this **new configuration will be then available to the other users**, who will be able to load it (either by discarding their draft or automatically, if the user did not have any draft).

ℹ Until you deploy your configuration, you are playing in your own playground. When you deploy the configuration, it will become visible and active for PSP and available to other users of Strategy Designer, who would be able to load it.

Advanced Configuration (Strategy Designer)

⚠ This section is **for advanced users only**. An error in configuration will result in Strategy Designer not loading correctly or not loading at all.

- Deployment and Installation (Strategy Designer)
- Upgrade to Latest Version (Strategy Designer)
- Configuration Options (Strategy Designer)
- Exclude Out-of-the-box PSP Parameters (Strategy Designer)
- Include Additional PSP Parameters (Strategy Designer)
- Create Custom Blocks in Groovy (Strategy Designer)

Deployment and Installation (Strategy Designer)

Strategy Designer is deployed as **part of the Price Setting Package from the Pricefx PlatformManager Marketplace**. This is the preferred way because it also creates the necessary business role.

⚠ If you deploy it manually, make sure you have the **Price Setting Package version 2.1.3** or newer installed on your partition.

Manual Deployment

To manually deploy Strategy Designer, follow these steps:

1. Find the **number of the latest release** of the Strategy Designer. You can find it at the home of this guide.
2. Go to **Configuration** > **Advanced Configuration Options**.
3. Create a new entry named `pfxExternalApp_visual_configuration_strategy_designer`. The important part is the prefix `pfxExternalApp_`, the rest is up to you.
4. Copy and paste the following JSON as the entry's value:

```
{
  "name": "strategy-designer",
  "label": "Strategy Designer",
  "url": "https://apps.pricefx.com/visual-configuration/1-0-0/#
/strategy-designer",
  "businessRole": "StrategyDesigner",
  "configuration": {
  }
}
```

⚠ Notice the `url` parameter and the version number 1-0-0 in the URL. **Replace this number with the latest version**, only **use dashes** instead of dots. For example, 1.0.1 becomes 1-0-1.
This creates a default configuration for Strategy Designer. It can be further adjusted via the configuration options.

5. Create a **new business role** called *StrategyDesigner* and assign it the following permissions:
   - Master Data > Administer Company Parameters
   - Price Setting > Manage Calculation Logics
   - Price Setting > View LPG
   - Price Setting > View Price Lists
   - Administration > Data Integration
6. **Assign this business role** to the users who should be able to use Strategy Designer.
7. **Refresh** your browser.

After these steps, Strategy Designer should appear in the main menu under External Applications.

⚠ Make sure your JSON is syntactically correct. No trailing commas are allowed.

⛔ The StrategyDesigner role will give the users access to certain Pricefx features, which they otherwise might not have access to. This is the current limitation. Be careful who you give access to.

Upgrade to Latest Version (Strategy Designer)

Currently, the only way to upgrade your existing Strategy Designer configuration is to **manually update the** `url` **parameter** in the Advanced Configuration Options.

1. Find the **number of the latest release** of the Strategy Designer. You can find it at the home page of this guide.
2. Go to **Configuration > Advanced Configuration Options**.
3. **Open the entry** with your Strategy Designer configuration. It starts with **pfxExternalApp_**, e.g. `pfxExternalApp_visual_configuration_strategy_designer`.
4. **Update the** `url` **parameter** to point to the new version:
   Strategy Designer is deployed at `https://apps.pricefx.com/visual-configuration/{VERSION}/#/strategy-designer` where `{VERSION}` is the released version's number with dashes instead of dots.
   For example, version 1.0.1 is at URL `https://apps.pricefx.com/visual-configuration/1-0-1/#/strategy-designer`.
5. **Save** the entry.
6. **Refresh** your browser.
7. **Open** Strategy Designer again from the External Applications in the main menu.

Configuration Options (Strategy Designer)

Strategy Designer's additional configuration is stored in its `configuration` JSON in the `pfxExternalApp_visual_configuration_strategy_designer` entry in the Advanced Configuration Options. The value is a JSON object.

A default configuration looks like this:

```
{
  "name": "strategy-designer",
  "label": "Strategy Designer",
  "url": "https://apps.pricefx.com/visual-configuration/1-0-0/#
/strategy-designer",
  "businessRole": "StrategyDesigner",
  "configuration": {
    "advancedConfigurationStateEntry": "visual_configuration_psp_state",
    "stateLookupTableName": "StrategyDesignerState",
    "dependencyConfigurationTable": "DependencyConfiguration",
    "priceListLogicName": "IndependentPriceListLogic",
    "priceSettingDimensionsTable": "PriceSettingDimensions",
    "strategyDefinitionTable": "StrategyDefinition",
    "livePreviewDependencyLevel": "Global",
    "groovyLibraryName": "CustomPricingStrategiesLib",
    "customBlocksGroovyLibraryName": "StrategyDesignerCustomBlocks",
    "excludedParameters": [],
    "additionalParameters": {}
  }
}
```

You can only **specify the parameters you want to change**. Others will use their default values.

Values of name, label, url, and businessRole are needed for Unity in order to display Strategy Designer in the menu.

- **name** – Unique name among the external applications which will be part of the URL. Make sure it is unique among all `pfxExternalApp_` entries.
- **label –** Text of the menu item. You can safely change this.
- **url** – Points to the server where Strategy Designer is deployed.
- **businessRole** – Specifies the business role that the user has to have assigned to see the menu item.
- **configuration –** Application-specific configuration, in our case it **contains the configuration options for the Strategy Designer**.

Here is a list of options you can specify within the **configuration** object, with their default values and a description. A default value is used when you omit the option.

Option	Default Value	Description
`advancedConfigurationStateEntry`	`"visual_configuration_psp_state"`	Key in the Advanced Configuration Options **where the deployed state of the Strategy Designer is stored**. Change this if you want Strategy Designer to store its deployed state elsewhere, or load it from elsewhere.

`stateLookupTableName`	`"StrategyDesignerState"`	Company Parameter **where the draft state of the Strategy Designer is stored**.
`dependencyConfigurationTable`	`"DependencyConfiguration"`	PSP table where dependencies are configured. Unless you have a custom PSP deployment, you should not need to change this.
`priceListLogicName`	`"IndependentPriceListLogic"`	Name of the PSP's IndependentPriceListLogic. Unless you have a custom PSP deployment, you should not need to change this.
`priceSettingDimensionsTable`	`"PriceSettingDimensions"`	PSP table where dimensions are configured. Unless you have a custom PSP deployment, you should not need to change this.
`strategyDefinitionTable`	`"StrategyDefinition"`	PSP table where strategy definitions are configured. Unless you have a custom PSP deployment, you should not need to change this. This is **where the custom strategies are written when they are deployed**.
`livePreviewDependencyLevel` OBSOLETE	`"Global"`	Default dependency level for the Live Preview. It became obsolete in version 1.0.0 where you can pick the level yourself from a drop-down.
`groovyLibraryName`	`"CustomPricingStrategiesLib"`	Name of a Groovy Library which **contains the custom strategies' generated code**. It does not have to exist; it will be created during the first deployment.
`customBlocksGroovyLibraryName`	`"StrategyDesignerCustomBlocks"`	Name of a Groovy Library which contains definitions of custom blocks. Learn how to define custom blocks in Create Custom Blocks in Groovy (Strategy Designer).
`excludedParameters`	`[]`	List of default PSP parameters you want to exclude. Specifying them will hide the blocks from the workspace. For details see Exclude Out-of-the-box PSP Parameters (Strategy Designer).
`additionalParameters`	`{}`	List of additional PSP parameters you want to see as blocks. Typically, you would need to specify any additional parameters you created in your customized PSP logic. For details see Include Additional PSP Parameters (Strategy Designer).

Exclude Out-of-the-box PSP Parameters (Strategy Designer)

Strategy Designer offers you to use some of the out-of-the-box parameters calculated by PSP as blocks in your strategies. But if they are not configured properly in the PSP, they will not work and will result in

errors in Strategy Designer. In such a case, it is better to hide them from the toolbox so the users do not use them.

If you do not want to use them, you can specify them in the list of excluded parameters and Strategy Designer will hide them from the toolbox.

The following PSP parameters are supported:

- PRODUCT_COST - comes from the standard Cost Lookup
- BASE_PRICE - comes from the standard Actual Price Lookup
- PLUS_FOR_PRODUCT_ABSOLUTE - comes from the Cost Plus Lookup
- PLUS_FOR_PRODUCT_PERCENTAGE - comes from the Cost Plus Lookup
- PRICE_INCREASE_ABSOLUTE - comes from the Price Increase Lookup
- PRICE_INCREATE_PERCENTAGE - comes from the Price Increase Lookup

Here is an example of the configuration object that hides the last four parameters:

```
"configuration": {
  "excludedParameters": [
    "PLUS_FOR_PRODUCT_ABSOLUTE",
    "PLUS_FOR_PRODUCT_PERCENTAGE",
    "PRICE_INCREASE_PERCENTAGE",
    "PRICE_INCREASE_ABSOLUTE"
  ]
}
```

You can also easily hide all PSP parameters like this:

```
"configuration": {
  "excludedParameters": "ALL"
}
```

Include Additional PSP Parameters (Strategy Designer)

EXPERT LEVEL

In case your custom price list logic uses some additional parameters (typically configured for Custom Engines), you can add them as blocks using the `additionalParameters` option.

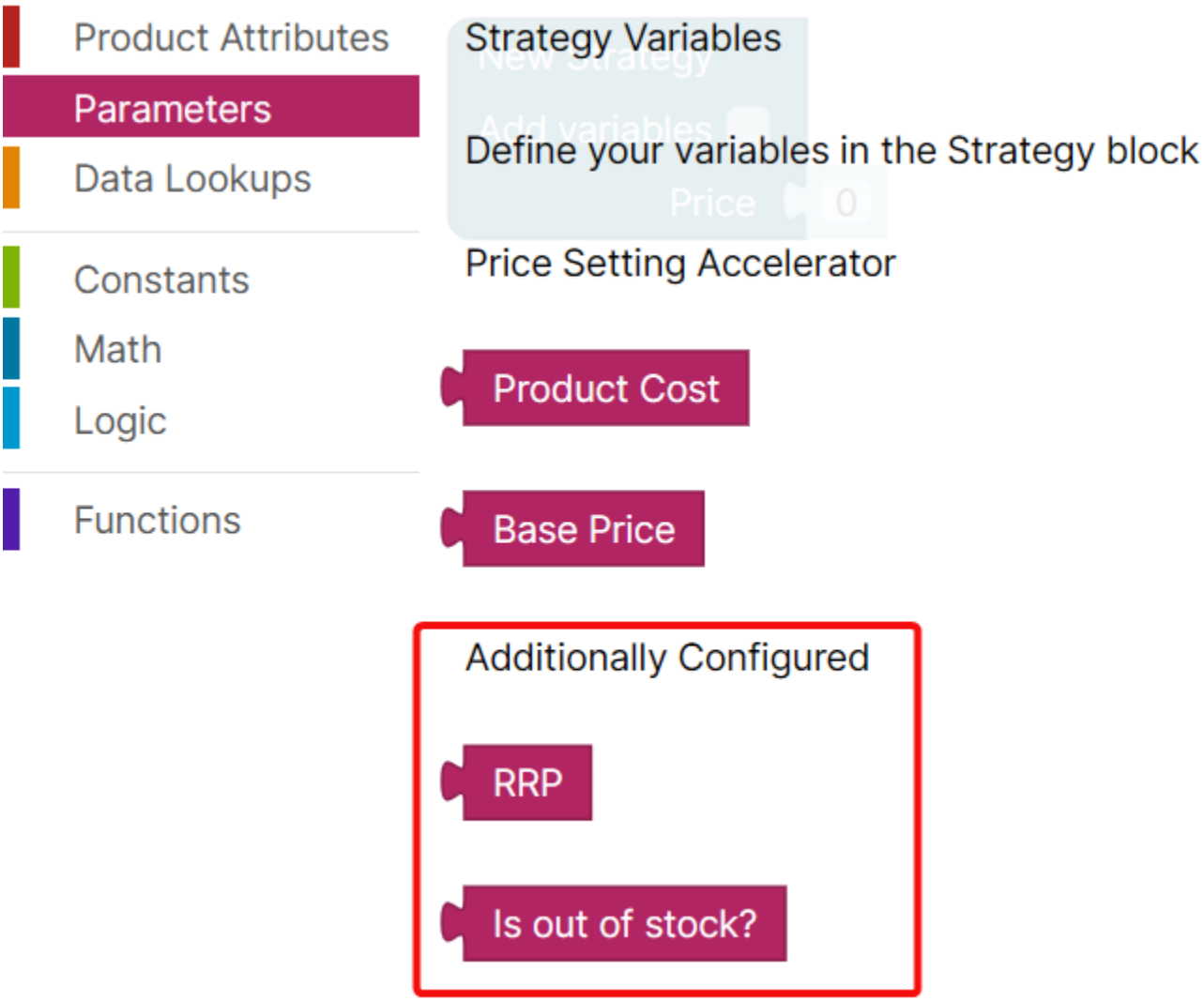
The `additionalParameters` parameter is an object where each key is the PSP's identifier of the parameter and the value is an object with a `label` and a `type`. The `label` will become the text displayed on the corresponding block in the toolbox and the `type` will become the return type of that block. Supported types are `Number`, `Boolean`, and `Text`.

Say we have two additional parameters called 'RRP' which is a number, and 'IS_OUT_OF_STOCK' which is a boolean value (true or false). This is how you would make them available in the Strategy Designer's toolbox:

```
"configuration": {
  "additionalParameters": {
    "RRP": {
      "label": "RRP",
      "type": "Number"
    },
    "IS_OUT_OF_STOCK": {
      "label": "Is out of stock?",
      "type": "Boolean"
    }
  }
}
```

Each key in the `additionalParameters` object has to be unique. Make sure there are no syntax errors in the JSON object.

The parameters will be available in the toolbox under the Parameters category:



Create Custom Blocks in Groovy (Strategy Designer)

EXPERT LEVEL

If blocks provided by Strategy Designer out-of-the-box are not enough, you can define your own using Groovy.

Create Groovy Library

Follow these steps:

1. Create a Groovy Library and name it for example 'StrategyDesignerCustomBlocks' (write down this name for later).
2. In the library, create the first element and name it **Meta**. Set its **Display Mode** to *Everywhere*. More on the Meta element later.
3. Create other elements (at least one) which will contain the functions your custom blocks will call.

Meta Element

The Meta element **must return an array of objects**, each object representing a block you want to create in Strategy Designer.

Each block has a label and can contain parameters (block inputs where other blocks can be plugged in). The block will then call a specified custom Groovy function located in another element of this library, passing the values of the user-provided parameters to it.

The block object has the following structure:

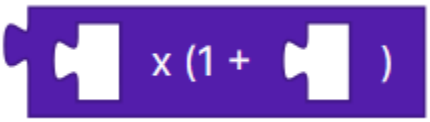
Property	Type	Required?	Description
`element`	String	Required	Location of the Groovy function which the block calls.
`function`	String	Required	Name of the function in the element specified by the `element` property, which the block calls.
`params`	List	Optional	List of parameter objects specifying the inputs of the block. Each parameter object has the following properties: • `name` (String) – Unique name of the parameter. • `type` (String) – Type of the parameter, either *number*, *string*, or *option*. • `default` – Default value supplied if the user does not provide the input; its type must correspond with the `type` property. • `options` – Specified only if the `type` is *option*. It is **either a list of values, or a map** where the key is the option's value and value is the option's label.
`label`	String	Required	

			Label rendered on the block. It can contain parameter placeholders like `{1}`, `{2}`, etc. If `params` are specified, each parameter will be rendered at the position of the corresponding placeholder, so the first parameter will be rendered at the `{1}` placeholder, the second at the `{2}` placeholder, and so on. If no placeholders are specified, the parameters will be all added at the end of the block.
`color`	Integer or String	Optional	Color of the block. It can be either an integer value (0 to 360) representing HSV hue value, or a #rrggbb String representing a web color. If no color is specified, the block will have the default color of the Functions toolbox category.
`tooltip`	String	Optional	Tooltip of the block which is displayed when the user hovers the pointer over the block. If empty, no tooltip will be displayed.
`returns`	String, either **number** or **string**	Required	Return type of the block. It should correspond with the type of the return value from the function defined by the `element` and `function` parameters.

Examples of Custom Blocks

Arithmetic Block

Let's say we want to create an arithmetic block for calculating a percentage mark-up that will look like this:



1. Create the **Meta** element and return a list with a single block configuration object from it, like this:

```
return [
  [
    "element" : "Math",
    "function": "markupPct",
    "label"   : "{1} x (1 + {2})",
    "params"  : [
            ["name": "base", "type": "number"],
            ["name": "markup", "type": "number", "default": 0]
    ],
    "returns" : "number"
  ]
]
```

   Notice the label containing the placeholders that make the inputs render at the correct places.
2. Create a **Math** element and in it create a function called **markupPct** with two parameters, like this:

```
BigDecimal markupPct(BigDecimal base, BigDecimal markup) {
  if (base == null) {
```

```
    throw new Error("The base cannot be empty");
  }
  if (markup == null) {
    return base;
  }
  return base * (1 + markup)
}
```

You can name the element and function whatever you want as long as it is the same name as specified in the `element` and `function` properties in the block configuration object.

If you have multiple custom blocks, return all of them in the list returned by the Meta element.

You can have as many or as few elements with your functions as you want. A good practice is to group your functions in elements with a similar purpose, like Math, Text or Competition Data.

Block with Drop-down

Let's say we want to create a block that will retrieve an aggregation of prices of a selected competitor. The block should look like this:

1. Create the **Meta** element and return a list with a single block configuration object from it (or append the object to the existing list of objects), like this:

```
def competitorsIt = api.stream("PCOMP", "competitor",
["competitor"], true)
def competitors = competitorsIt.collect { it.competitor }
competitorsIt?.close();

def operatorOptions = [
        "MAX"  : "highest",
        "MIN"  : "lowest",
        "AVG"  : "average",
        "SUM"  : "total",
        "FIRST": "oldest",
        "LAST" : "latest"
]

return [
  [
    "element" : "Competition",
    "function": "competitorPrice",
    "status"  : "active",
    "label"   : "Get {1} price of competitor {2}",
    "params"  : [
```

```
            ["name": "operator", "type": "option", "options":
operatorOptions],
            ["name": "competitor", "type": "option", "options":
competitors]
    ]
  ]
]
```

Notice how we first retrieved the available options from the PCOMP table (line 1) and mapped it into an array of competitor names (line 2). We used this list of competitors in the `competitor` input parameter (line 22). Instead of a list, we could also feed it a map where the key is a competitor ID and the value is the competitor's name, if necessary. We took this approach with the `operatorOptions` (lines 5 and 22).

2. Create a **Competition** element and in it create a function called **competitorPrice** with two parameters, like this:

```
BigDecimal competitorPrice(String operator, String competitor =
null) {
    def filters = []
    if (!operator) {
        return null
    }
    if (competitor) {
        filters << Filter.equal("competitor", competitor)
    }
    def records = api.productCompetition(Filter.and(filters.
toArray()))
    records.forEach { api.trace 'Record', it.infoDate + ' | ' + it.
competitor, it.price }
    if (!records) {
        return null
    }
    switch (operator) {
        case "MAX":
            return records.max { it.price }?.price
        case "MIN":
            return records.min { it.price }?.price
        case "AVG":
            return records.sum { it.price } / records.size()
        case "SUM":
            return records.sum { it.price }
        case "FIRST":
            return records.sort { a, b -> a.infoDate <=> b.
infoDate }.first()?.price
        case "LAST":
            return records.sort { a, b -> a.infoDate <=> b.
infoDate }.last()?.price
        default:
            return null
```

```
    }
}
```

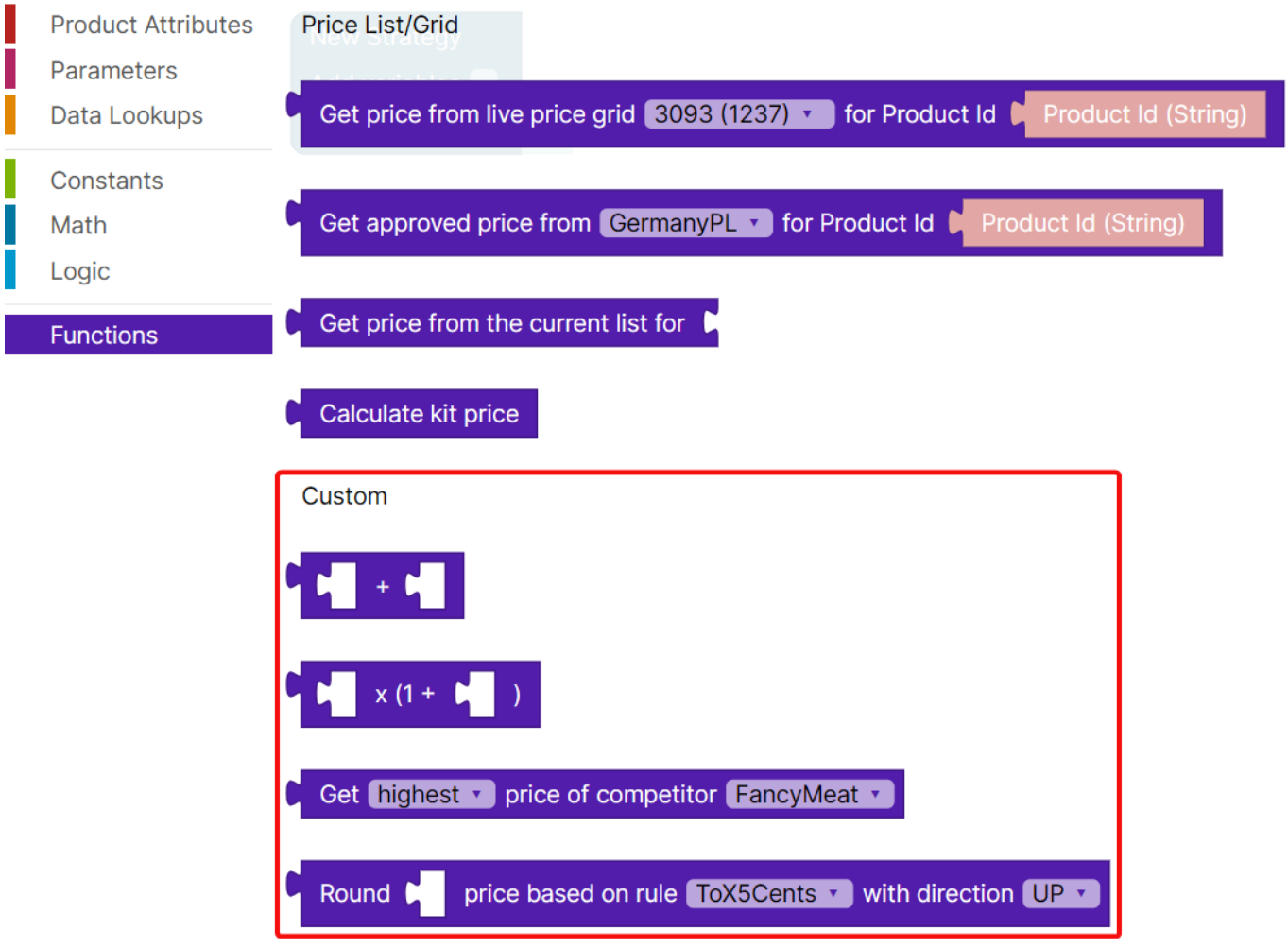
Update Configuration

When your Groovy library is configured, you just need to tell Strategy Designer that it should read it and create blocks from the definitions. You can do it by specifying the `customBlocksGroovyLibraryName` option and setting it to your library's name.

For example:

```
"configuration": {
  "customBlocksGroovyLibraryName": "StrategyDesignerCustomBlocks"
}
```

When you reload the Strategy Designer, it will show all the defined blocks in the Functions category in the Strategies toolbox:



Limitations (Strategy Designer)

Strategy Designer is being actively developed. Here is a list of its known limitations:

- **English only**. Other languages are not supported at the moment, even if you switch to another language in the Unity settings.
- **Pricing is supported only on the independent level**, the same as the Live Preview. Dependent levels are not supported at the moment.
- **Limited multi-user access**. Each user can have their own draft, but only one active deployed configuration can be stored. When deployed, you currently cannot go back.
- **Extended user permissions**. Because Strategy Designer needs to modify formulas and have access to master data and price lists & grids, it needs to be given certain permissions. The users will then gain access to certain Pricefx features, which they otherwise might not have access to.

Frequently Asked Questions (Strategy Designer)

Q: I cannot see Strategy Designer in the main menu. Why?
A: You probably do not have the StrategyDesigner business role assigned. If you have, please check the configuration options, maybe there is another business role specified that you need to have.

Q.: Strategy Designer gives me a 'Not authorized' error when I try to open it. What can I do?
A: Please check if the StrategyDesigner business role has all the needed permissions. You can find them here.

Q: Can I use Strategy Designer to modify other logics, such as Quotes or Rebates?
A: No, Strategy Designer works only in tandem with the Price Setting Package accelerator. No other use cases are supported at the moment.

Q: I have changed a data lookup but the change is not reflected in my strategy. Is that a bug?
A: You probably forgot to save the data lookup. You need to save it even when you rename it. Otherwise, the new name will not be reflected in your strategy.

Q: I have set up a range data lookup and want to change it. But the 'Enable range lookup' is disabled. Why?
A: If the lookup is already used in a strategy, you cannot change the range definition. You either need to delete the corresponding lookup block from your strategy, or create a new one by duplicating this one and saving it as a new data lookup.

Q: Where can I report bugs and suggest ideas?
A: We have a dedicated Jira project for this: PFVCFG (Pricefx Visual Configuration). When submitting a ticket, make sure to assign it to the Strategy Designer epic.

Workflow Designer Technical Guide

This guide is for Workflow Designer version **1.0.0** and newer, released as part of the **Paper Plane** release on June 25th, 2023. **The latest release as of July 12th is 1.0.1.**

Workflow Designer is a **visual configuration layer** built on top **of the Approval Workflow Package**. It is used to configure this accelerator visually without having to deep-dive into its many configuration tables. This means that it does not work without the accelerator.

Before you continue, please make yourself familiar with Accelerate Approval Workflow Package. Its functionality is not covered in this guide. Namely, learn about:

- Workflow types, steps, approvers & watchers, step conditions
- Structure of Quotes & Rebates (header, line items, folders)

⚠ Workflow Designer is released as a *preview version* and it is not generally available as part of Pricefx Unity yet. It should be used with caution.

In this section:

Prerequisites (Workflow Designer)

Workflow Designer aims to be intuitive enough for business users to use, however, there are some concepts you should get familiar with before diving into building your strategies.

These concepts are:

- Knowledge of how Pricefx stores data in Master data and Company Parameters.
- Basic knowledge of how workflows work in Pricefx and how Accelerate Approval Workflow Package builds on top of that.
- Knowledge of Basics of Drag & Drop Visual Configuration.

Installation (Workflow Designer)

Currently, Workflow Designer **needs to be deployed manually** on each partition where you want to use it.

Steps:

1. ⚠ Make sure the Approval Workflow Package is deployed on the partition **first**. Without it, Workflow Designer will not work.
2. Find the number of the latest release of Workflow Designer. You can find it at the home page of this guide.
3. Go to **Configuration** > **Advanced Configuration Options**.
4. Create a new entry with the name `pfxExternalApp_visual_configuration_workflow_designer` (the important part is the `pfxExternalApp_` prefix, the rest is up to you, only it has to be unique).
5. **Copy the following JSON** and paste it as the entry's value:

```
{
  "name": "workflow-designer",
```

```
  "label": "Workflow Designer (Preview)",
  "url": "https://apps.pricefx.com/visual-configuration/1-0-0/#
/workflow-designer",
  "businessRole": "WorkflowDesigner"
}
```

⚠ Notice the version number 1-0-0 in the `url` parameter. **Replace this with the latest version**, only **use dashes** instead of dots. For example, version 1.0.1 becomes 1-0-1 in the URL.

6. Go to **Access Admin** > **Business Roles Admin** and create a new business role called *WorkflowDesigne* and assign it the following privileges:
   - Master Data > Administer Company Parameters
   - Price Setting > Manage Calculation Logics
   - Administration > Data Integration
7. **Assign this business role** to your user.
8. **Refresh** the page and you should see Workflow Designer in the main menu under the External Apps.

Limitations

The current version of Workflow Designer is just a preview. Although it is fully usable, it has only a very limited feature set. The main limitations at the moment are:

- The workflow configuration is not read from the partition after you save it. Every time you start Workflow Designer, **you start from scratch**.
  To load the workflow configuration, you have to export it, save it on your device, and import it later.
- Workflow Designer does not tell you **what fields** you have in your Quote, Rebate, or Promotion. You need to know them yourself.

These limitations are temporary and should be resolved when Workflow Designer is released to the public.

Design Your Workflow (Workflow Designer)

The workflows are designed and deployed in three steps:

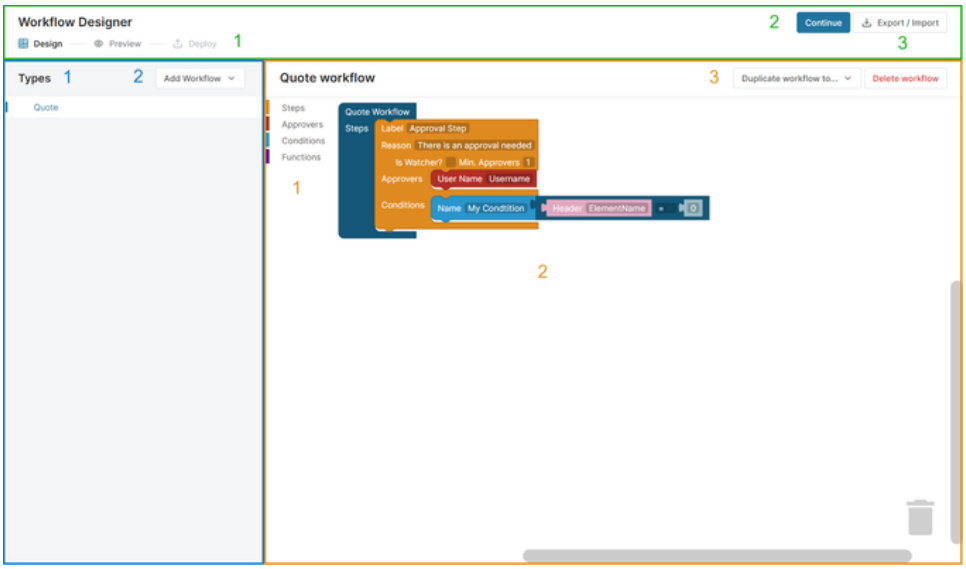
1. **Design** - Create, import or export workflows of different types using the drag & drop interface.
2. **Preview** - Check which workflow types should be deployed and preview the resulting configuration data if needed.
3. **Deploy** - Save the configuration data to the partition.

User Interface Overview

Here is the overview of the user interface of the Design step. If you have not read Basics of Drag & Drop Visual Configuration, now is the time.

Header

1. Interactive stepper which shows the current step. You can click the steps to navigate between them.



2. Continue button which takes you to the next step.
3. Export / Import function. It opens a side panel with options.

List of Workflows

1. List of configured workflow types.
2. Add Workflow button which allows you to add a new workflow type.

Workspace

1. Toolbox with available blocks divided into multiple categories.
2. Workspace where you connect the blocks.
3. Duplicate and Delete buttons:
   a. You can duplicate the current workflow type and create or overwrite a different one.
   b. You can delete the current workflow type.

Managing Workflows

When you first open Workflow Designer, there are no workflows configured. You can start creating a workflow by clicking the **Add Workflow** button or by **importing** a JSON file with a configuration.

Currently, there are these **workflow types** supported:

- Quote
- Price Grid Item
- Price List
- Rebate Agreement
- Rebate Record

The list of workflow types on the left side of the screen shows the workflow types which are configured. If a workflow type is not listed, it is not configured.

To configure a workflow type, click the **Add Workflow** button and choose the requested workflow type from the drop-down. Workspace, Toolbox, and Blocks will appear with a workflow with one dummy workflow step.

You can duplicate a workflow configuration to another workflow type by clicking the **Duplicate workflow to** button and selecting the target workflow type. Please note that if the target workflow type already has a configuration, it will be overwritten.

You can delete a workflow type's configuration by clicking the **Delete** button. This action cannot be undone.

Export / Import

> ⚠ Currently, **exporting your data** and saving it on your device is **the only option to preserve your workflows** in Workflow Designer. Make sure to do it so you can import them next time you want to work with them.

To export workflows:

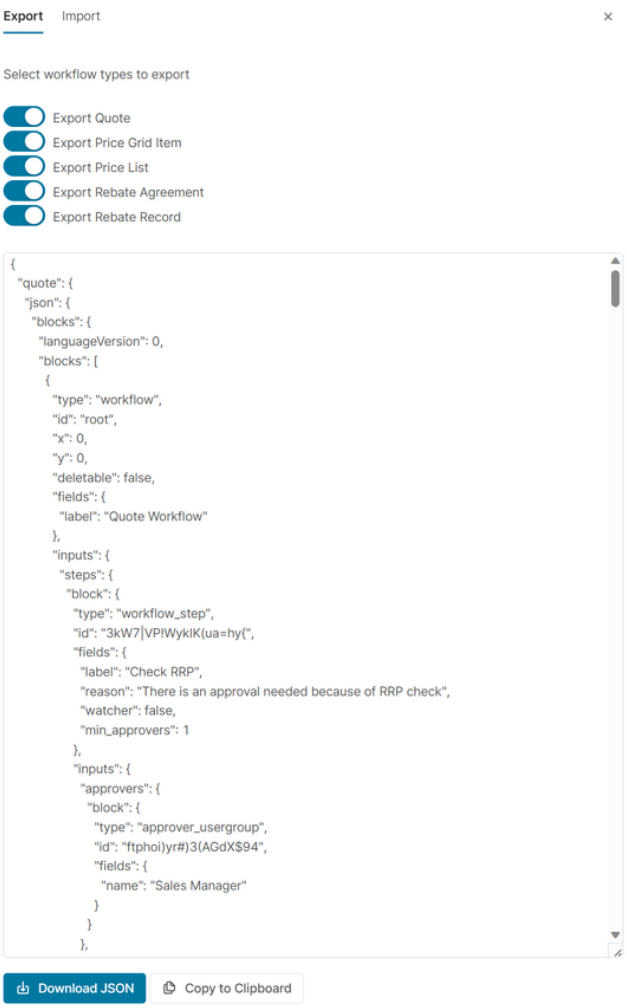
1. Open the Export / Import panel by clicking the **Export / Import** button and select the Export tab.
2. Select the workflow types you want to export.
3. Click the **Download JSON** button to download a JSON file which you can save on your device, or click **Copy to Clipboard** to copy the JSON to the system clipboard.

To import a workflow:

1. Open the Export / Import panel by clicking the **Export / Import** button and select the Import tab.
2. Paste the JSON either from the previously saved file or from the system clipboard.
3. Select the workflow types you want to import.
4. Click the **Import JSON** button to import the selected workflow types.

The **Import JSON** button will only be enabled if no errors are detected in the pasted JSON file. If there are errors, you will see an error message.

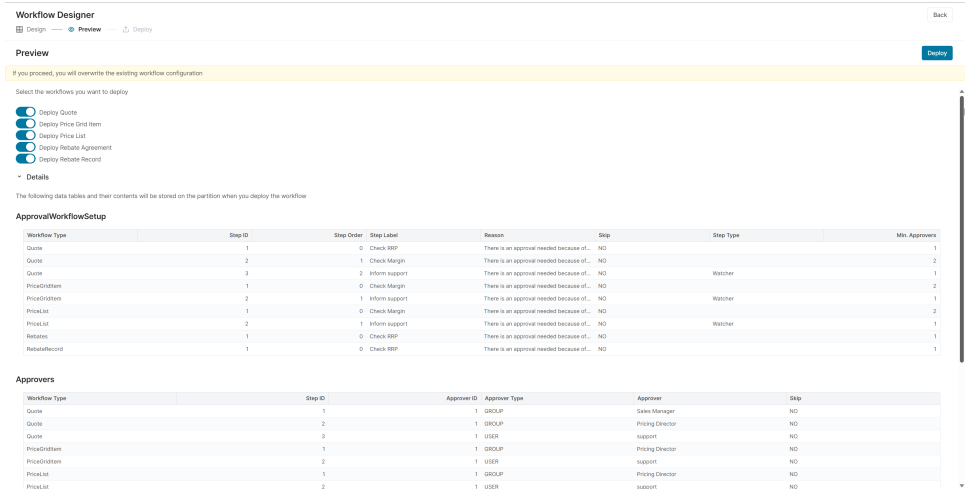
The system can detect some errors in the JSON file but not all of them. Make sure the JSON is syntactically correct and do not change it unless you know what you are doing.



Deploy to Partition (Workflow Designer)

Preview

When you are satisfied with your workflows, you can click the Preview step or click the **Continue** button; both take you to the Preview page.



On this page, you can:

- **Select the workflow types** you want to deploy.
- **Expand Details** and see the data that the tables of the Approval Workflow Package will be populated with.
- Deploy the workflows by clicking the **Deploy** button.

- Go back by clicking the **Back** button or the Design step and continue adjusting your workflows.

> ⚠ Your current workflows (related Company Parameter tables) will be completely overwritten when you run the deployment.

> ⛔ If there are no workflows configured and you click the **Deploy** button, you will essentially clear all the configuration.

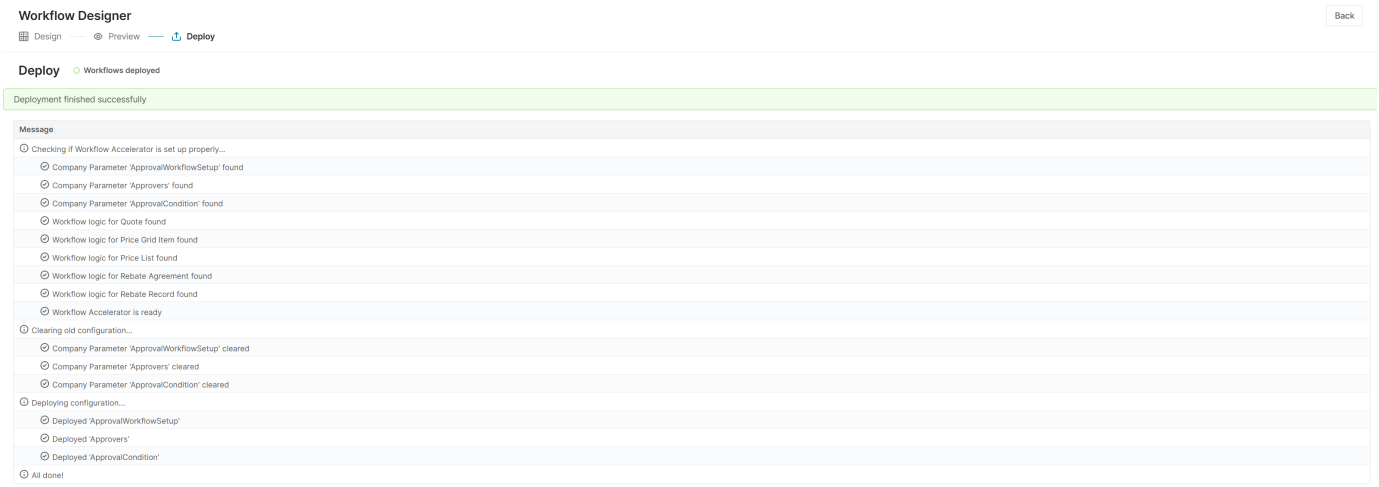
Deployment

When you click the **Deploy** button, you will be taken to the Deploy page where you can see the status of the deployment.

If the deployment finished successfully, you will get a success message.

From this screen, you can go back to the previous steps or you can close Workflow Designer if your work is done.

> ⚠ **When you close the Workflow Designer, it will not remember your workflow configuration**. The only way to preserve your workflows is to use the Export function, store the configuration on your local hard drive and import it the next time when you open the application.



If any errors were encountered during the deployment, you should **check the error messages** and act according to the information you receive. The errors will mostly be related to these areas:

- You do not have **sufficient privileges** to deploy the Company Parameter tables. In this case, check that the WorkflowDesigner role has all the necessary permissions assigned.
- There are **missing Company Parameter tables or Workflow Logics**. In this case, contact the Support team so they can check the configuration of the Approval Workflow Package Accelerator.