

PIMCORE INSPIRE' 21

The Data Director Bundle



Jan
Walther

blackbit

digital Commerce



Who?



- Jan Walther @ Blackbit
- Lead Developer

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018
 - Github: BlackbitNeueMedien

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018
 - Github: BlackbitNeueMedien
 - Most Valuable Pimconaut of the Year 2019

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018
 - Github: BlackbitNeueMedien
 - Most Valuable Pimconaut of the Year 2019
- Pimcore Community Contribution Rally 2020 Winner

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018
 - Github: BlackbitNeueMedien
 - Most Valuable Pimconaut of the Year 2019
- Pimcore Community Contribution Rally 2020 Winner
- Pimcore Contributor of the Month July 2021

Who?



- Jan Walther @ Blackbit
- Lead Developer
- Pimcore Core Contributor since 2018
 - Github: BlackbitNeueMedien
 - Most Valuable Pimconaut of the Year 2019
 - Pimcore Community Contribution Rally 2020 Winner
 - Pimcore Contributor of the Month July 2021
- Specialist for imports, exports and automation with Pimcore

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore
- In development since 2013, evolved with real user requirements to a battle-proven solution

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore
- In development since 2013, evolved with real user requirements to a battle-proven solution
- Successfully in use to
 - Import 8 million products and update 300K daily on a single Pimcore system

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore
- In development since 2013, evolved with real user requirements to a battle-proven solution
- Successfully in use to
 - Import 8 million products and update 300K daily on a single Pimcore system
 - Connect ERP systems, Webshops, InDesign and other external systems to Pimcore

Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore
- In development since 2013, evolved with real user requirements to a battle-proven solution
- Successfully in use to
 - Import 8 million products and update 300K daily on a single Pimcore system
 - Connect ERP systems, Webshops, InDesign and other external systems to Pimcore
- Used by > 100 clients

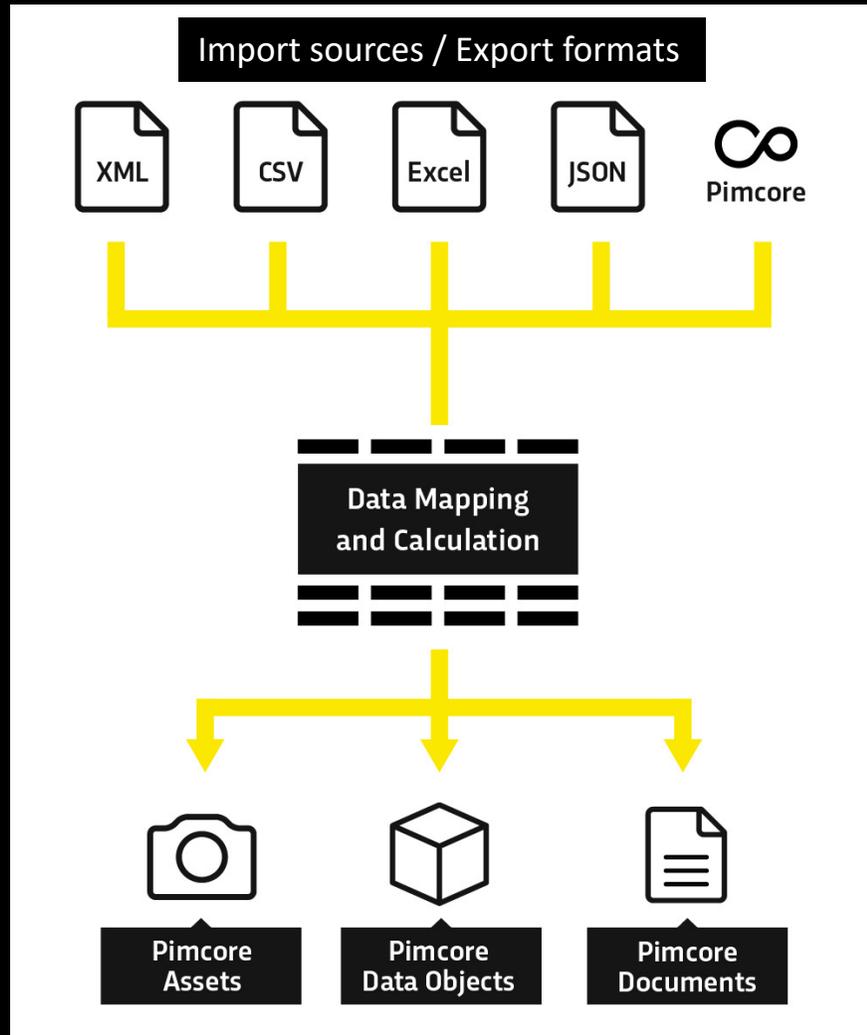
Data Director?

- Bundle for efficient connection of external systems to Pimcore (imports + exports)
- Automation of data control within Pimcore
- In development since 2013, evolved with real user requirements to a battle-proven solution
- Successfully in use to
 - Import 8 million products and update 300K daily on a single Pimcore system
 - Connect ERP systems, Webshops, InDesign and other external systems to Pimcore
- Used by > 100 clients

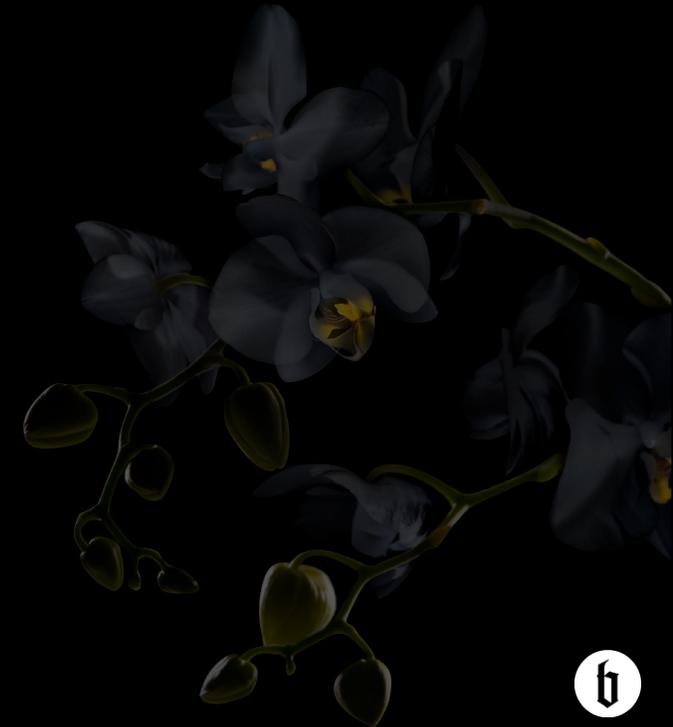


And lots more through our collaborations with agencies and Pimcore partners

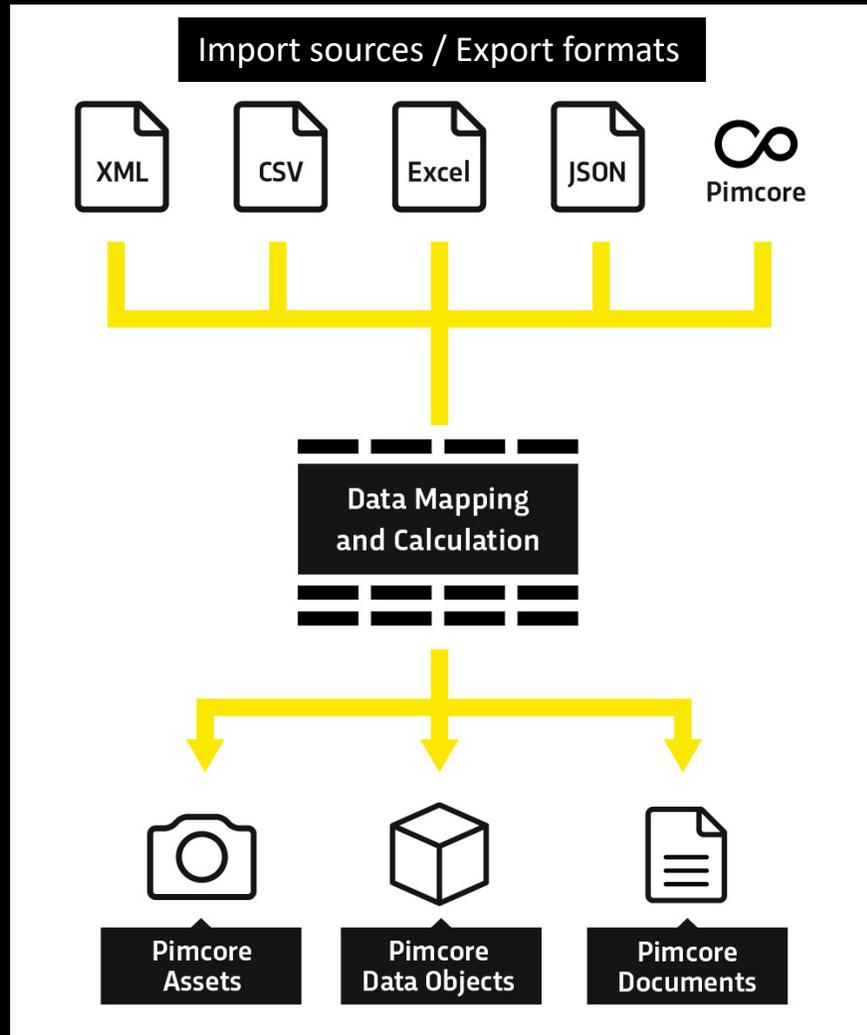
Data Director – Process flow



Setup of imports:

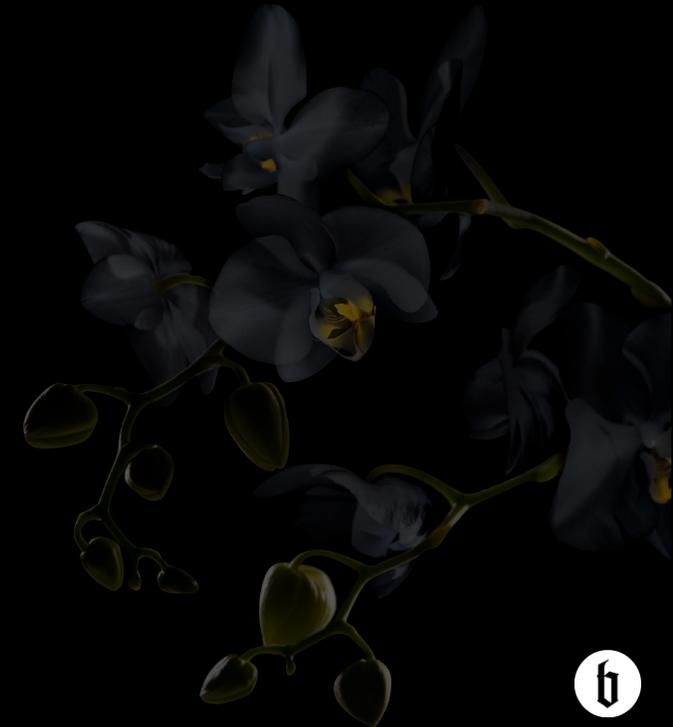


Data Director – Process flow

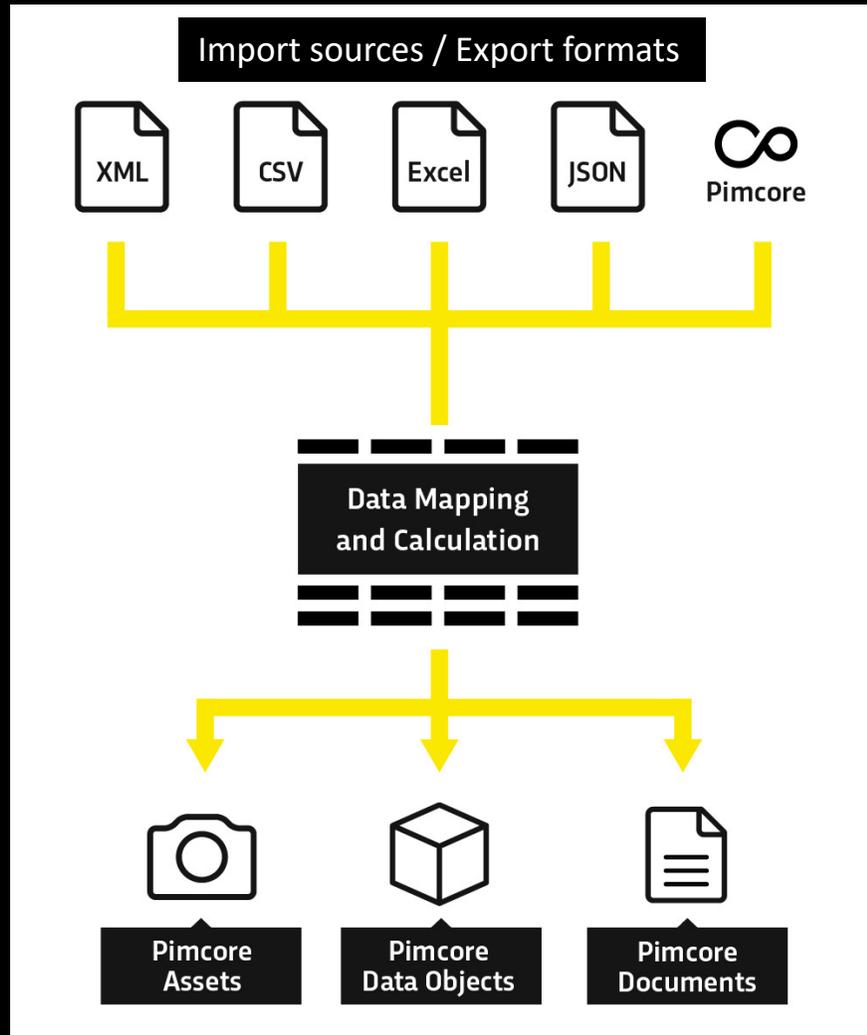


Setup of imports:

1. Extract raw data from data source

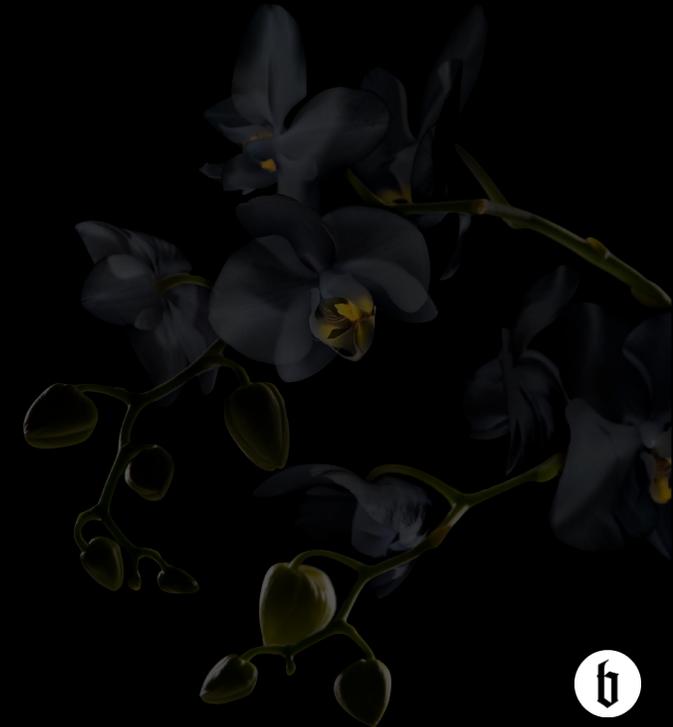


Data Director – Process flow

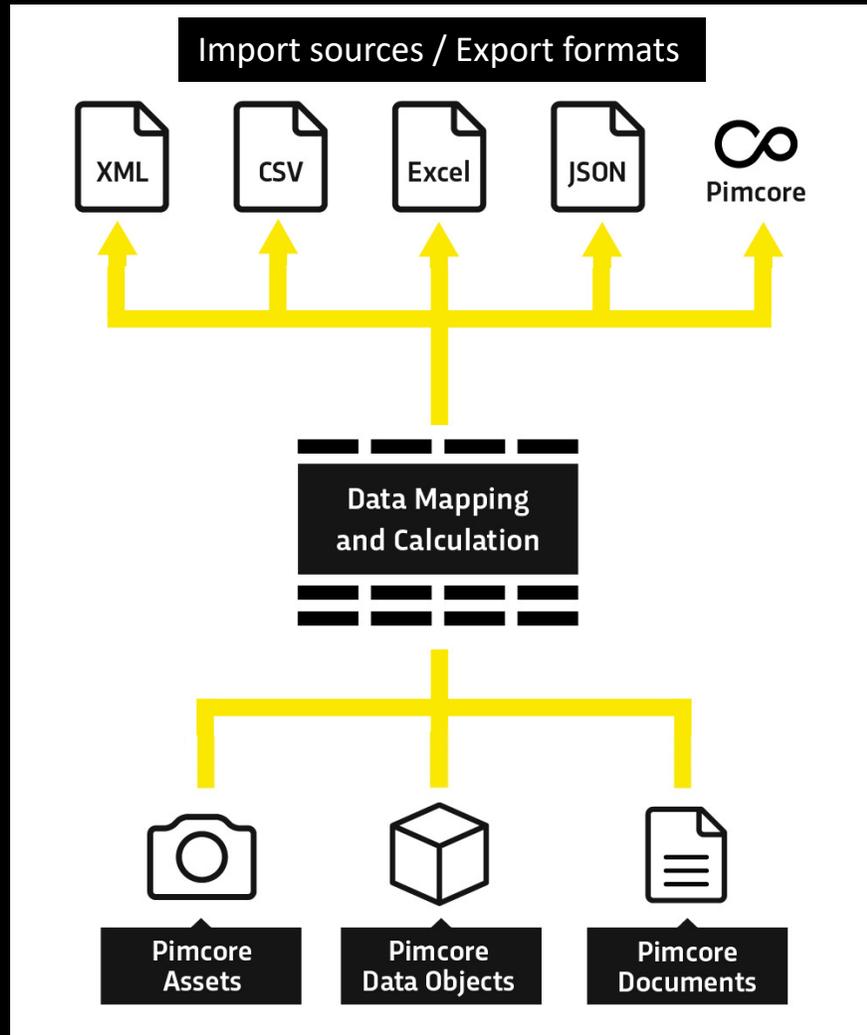


Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields



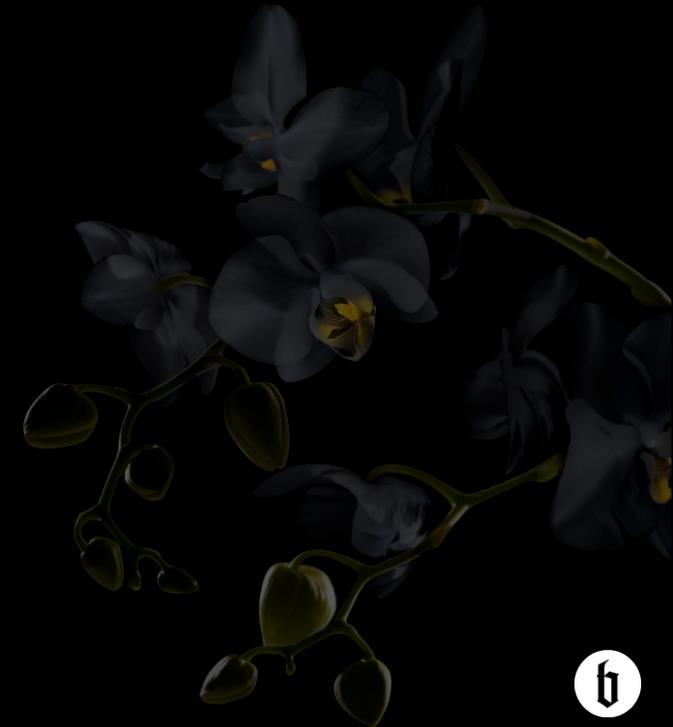
Data Director – Process flow



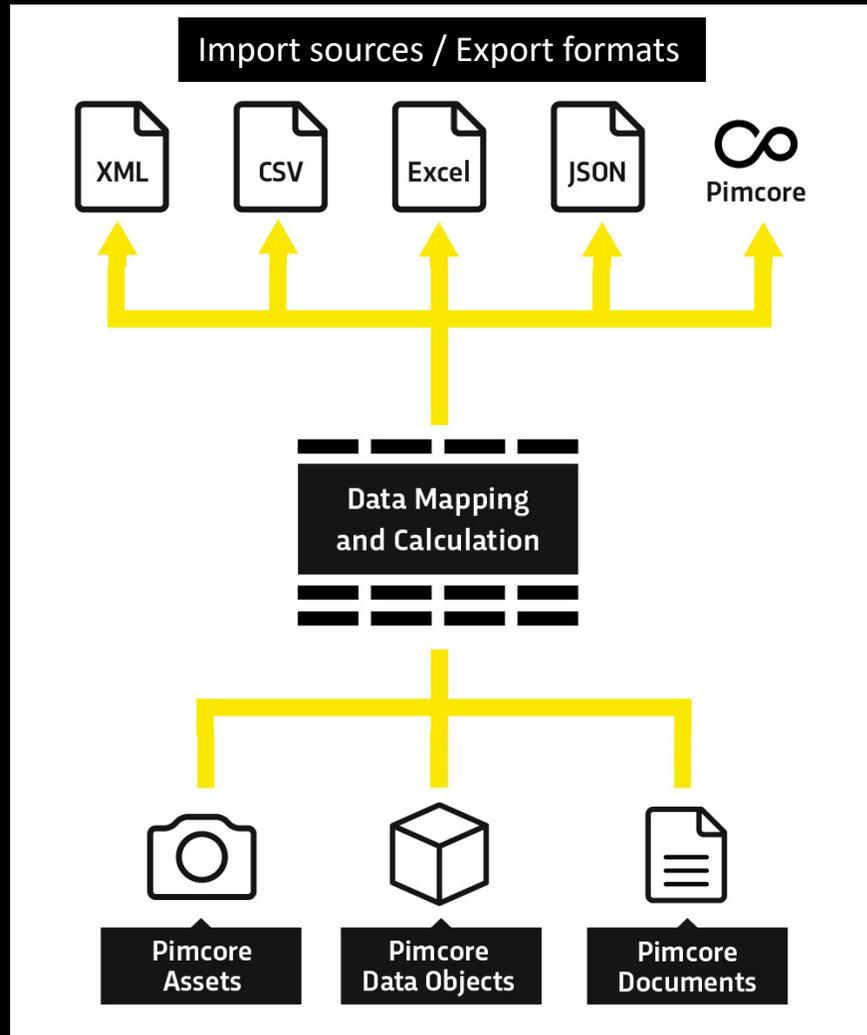
Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields

Setup of exports:



Data Director – Process flow

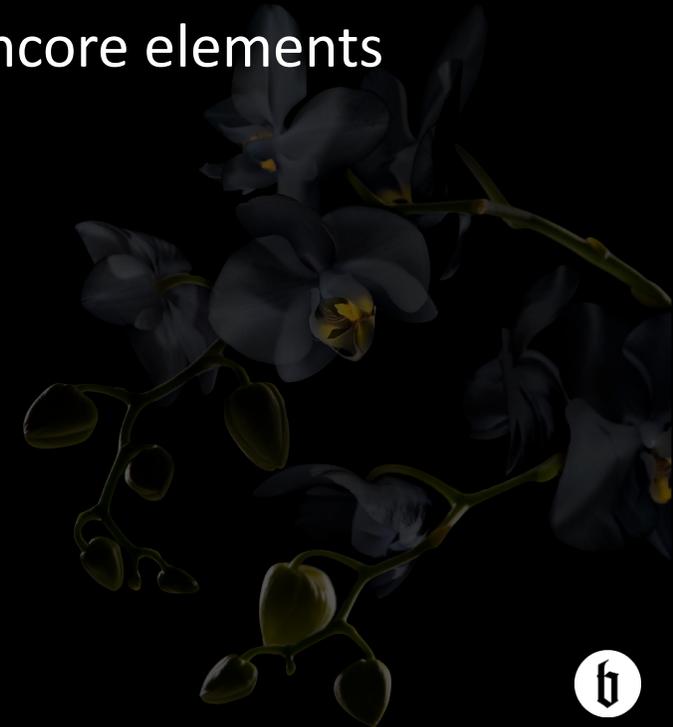


Setup of imports:

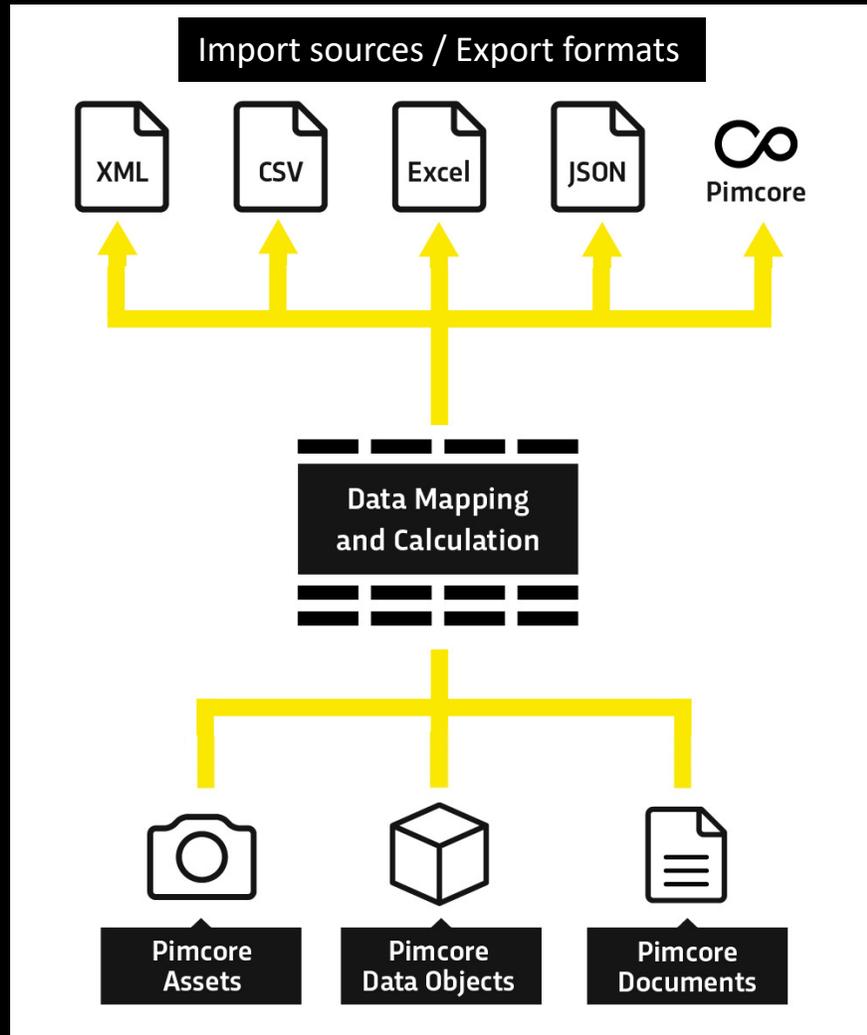
1. Extract raw data from data source
2. Map raw data fields to target class fields

Setup of exports:

1. Extract raw data from Pimcore elements



Data Director – Process flow



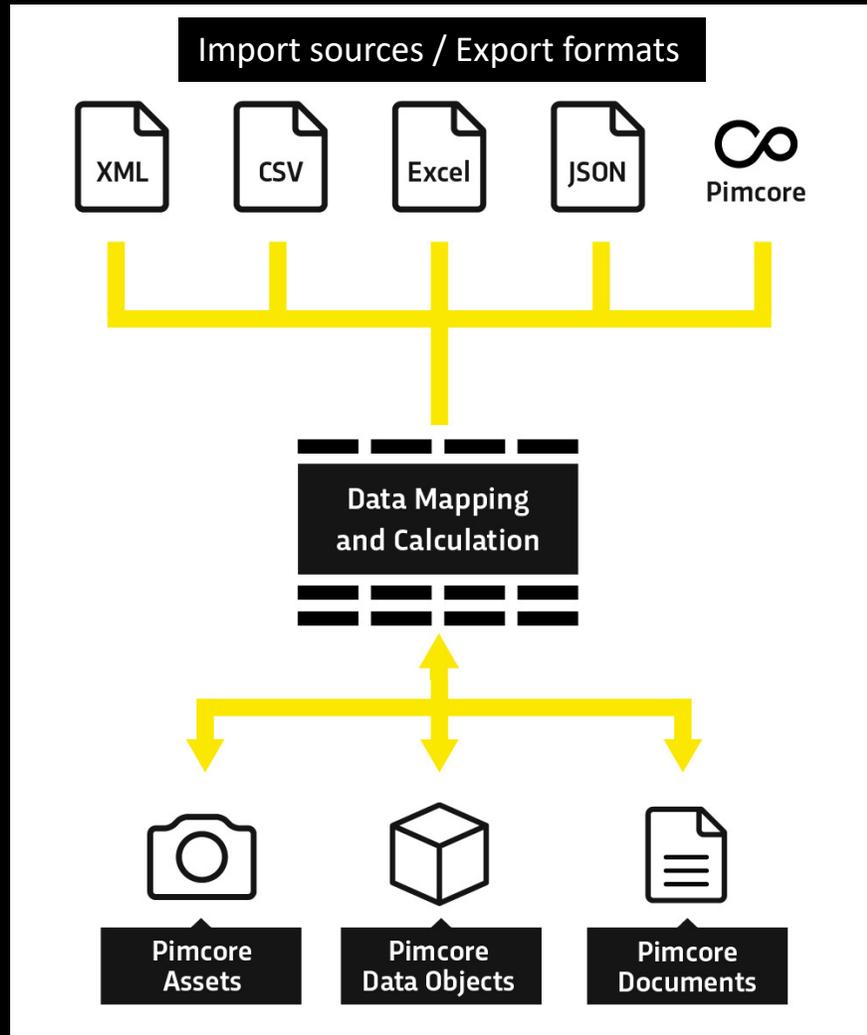
Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields

Setup of exports:

1. Extract raw data from Pimcore elements
2. Map raw data fields to fields of export document

Data Director – Process flow



Setup of imports:

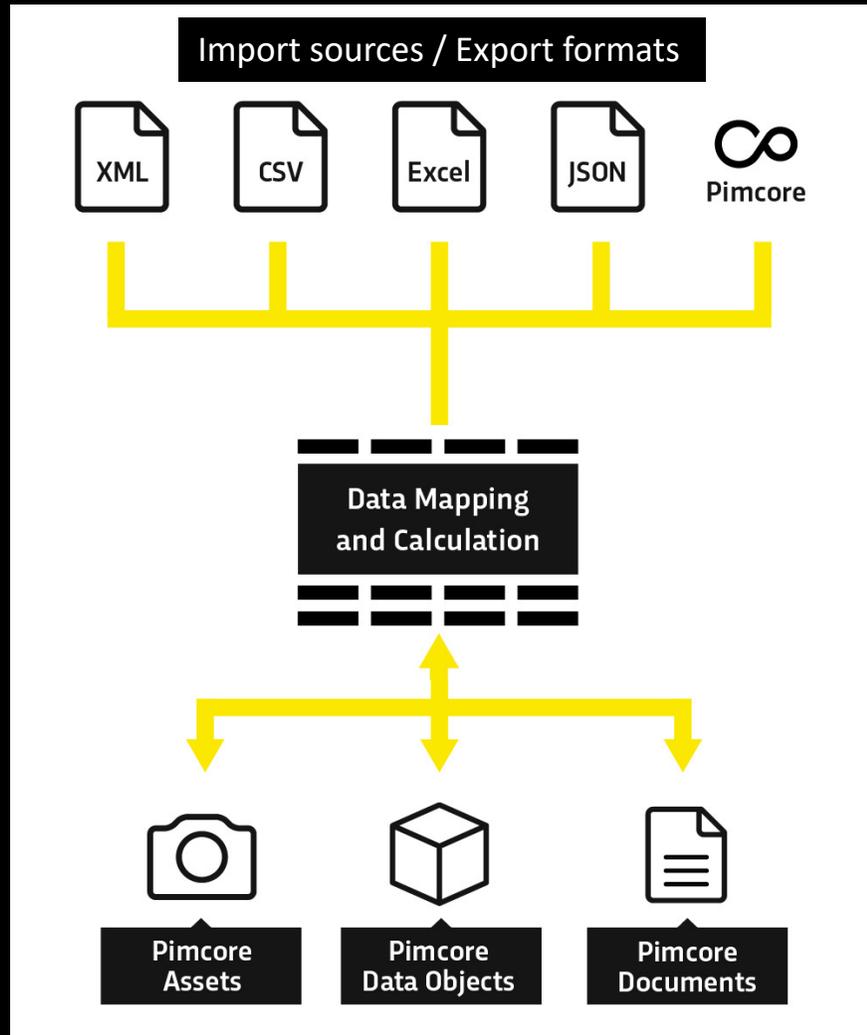
1. Extract raw data from data source
2. Map raw data fields to target class fields

Setup of exports:

1. Extract raw data from Pimcore elements
2. Map raw data fields to fields of export document

Setup of automation:

Data Director – Process flow



Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields

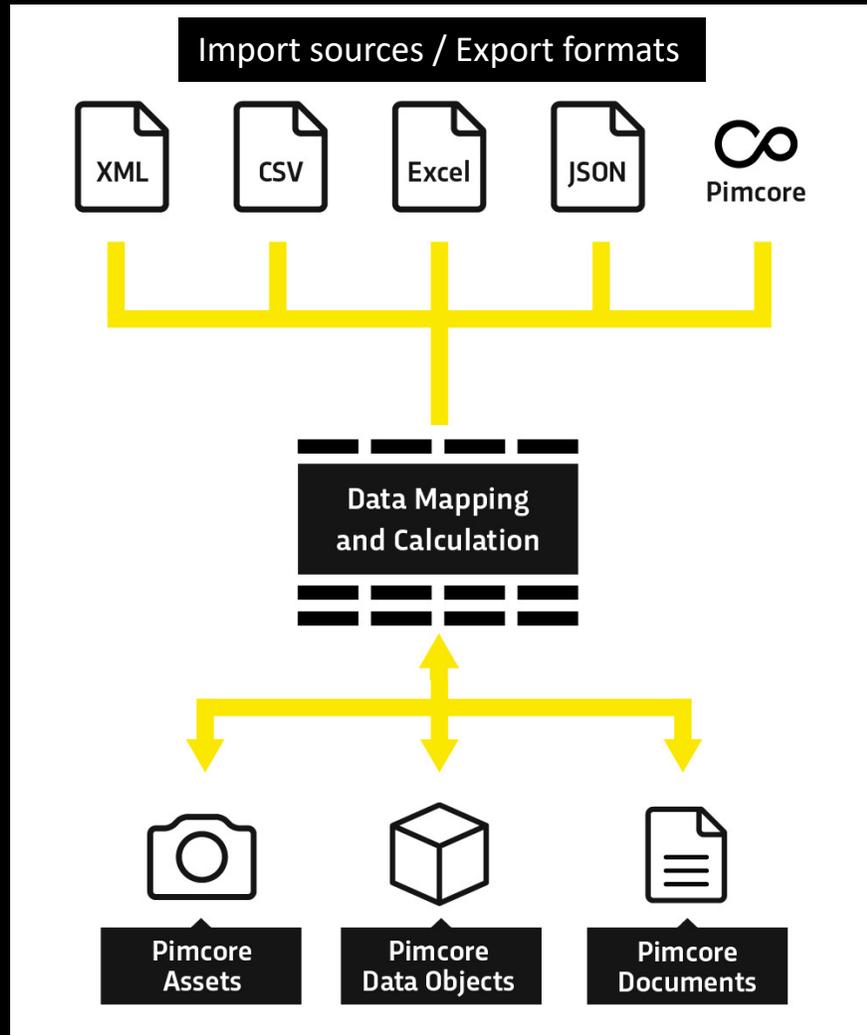
Setup of exports:

1. Extract raw data from Pimcore elements
2. Map raw data fields to fields of export document

Setup of automation:

1. Extract raw data from Pimcore elements

Data Director – Process flow



Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields

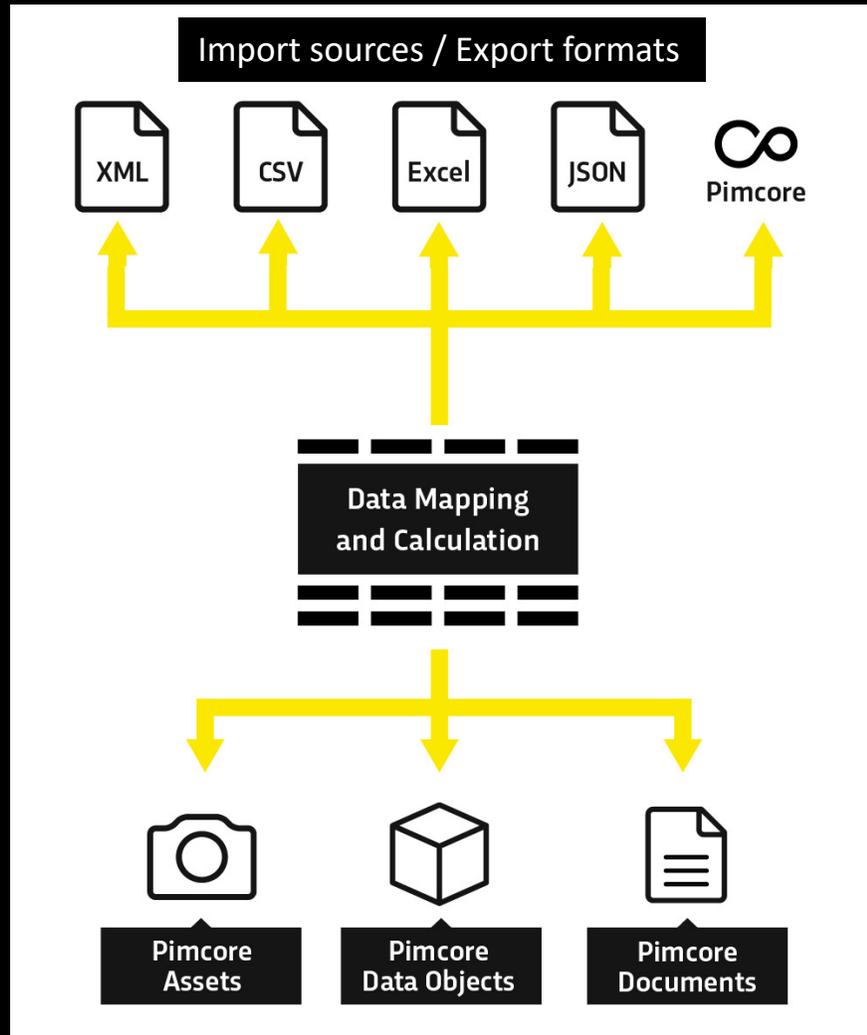
Setup of exports:

1. Extract raw data from Pimcore elements
2. Map raw data fields to fields of export document

Setup of automation:

1. Extract raw data from Pimcore elements
2. Map raw data fields to target class fields

Data Director – Process flow



Setup of imports:

1. Extract raw data from data source
2. Map raw data fields to target class fields

Setup of exports:

1. Extract raw data from Pimcore elements
2. Map raw data fields to fields of export document

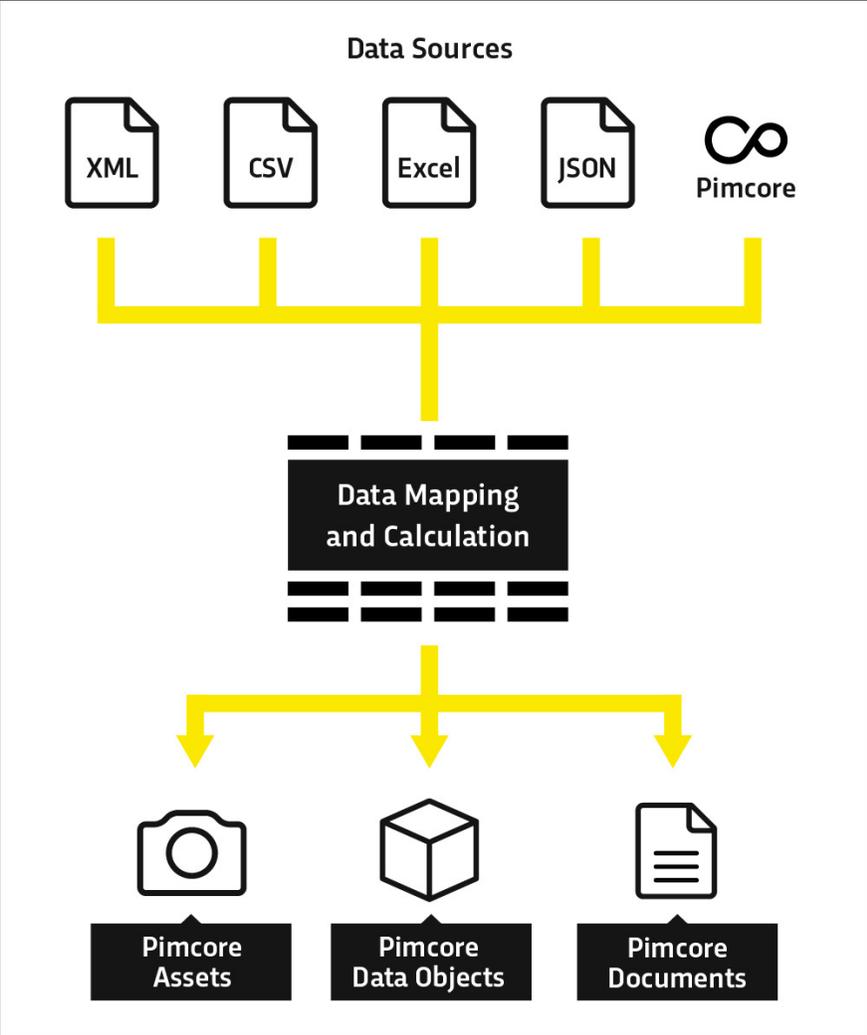
Setup of automation:

1. Extract raw data from Pimcore elements
2. Map raw data fields to target class fields

ALL THE SAME!

UNIFIED INTERFACE FOR ALL OPERATIONS

Live Demo Import



blackbit

digital Commerce

There are plenty of other fish in the sea

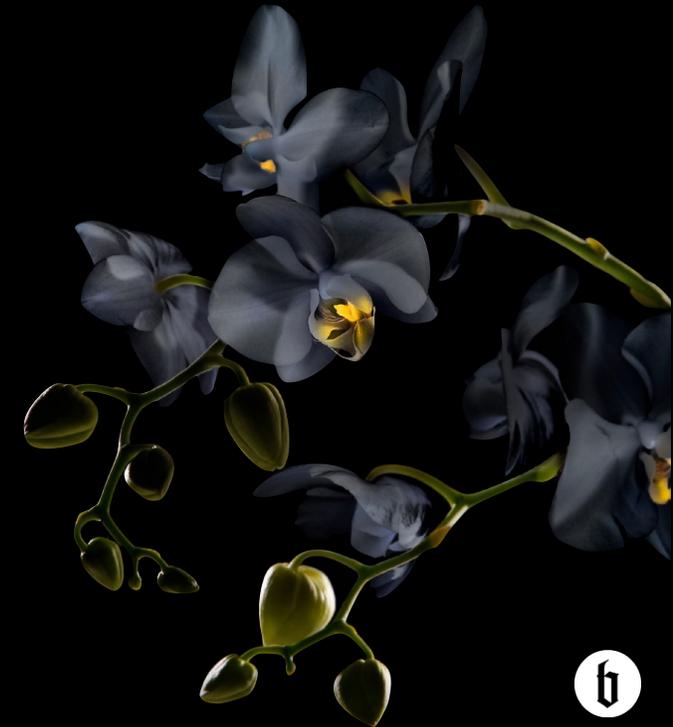
Advantages of Data Director compared to other bundles

Imports



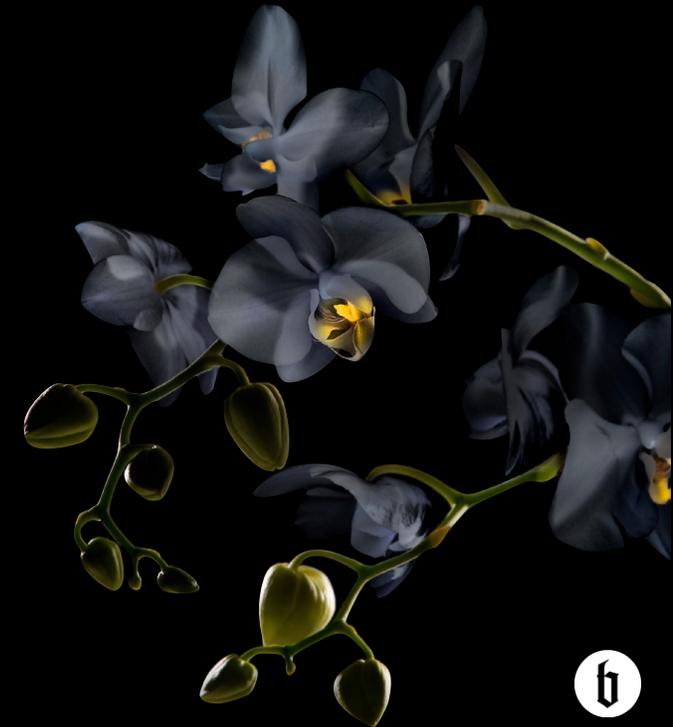
Flexible import resources

- Pimcore assets or filesystem:
 - Files



Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders



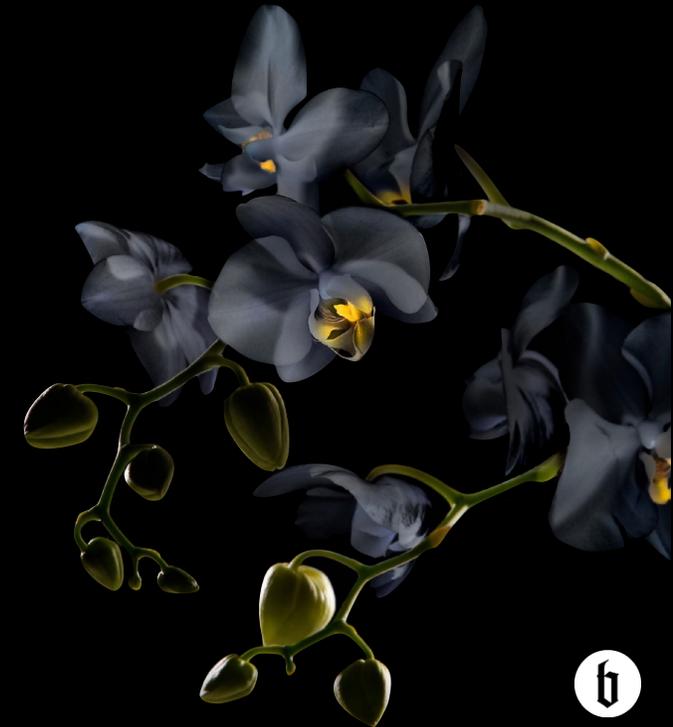
Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders
 - Glob expressions (/folder/*.xml)



Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders
 - Glob expressions (/folder/*.xml)
 - ZIP files



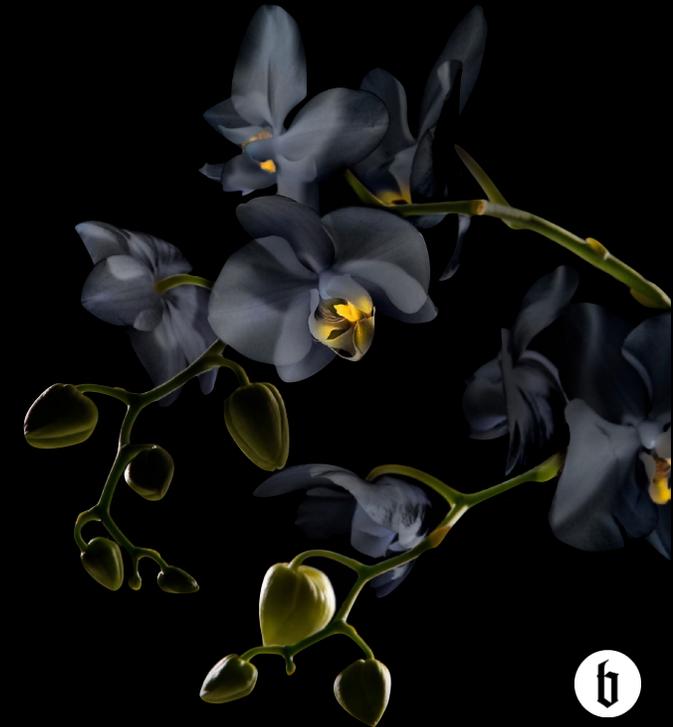
Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders
 - Glob expressions (/folder/*.xml)
 - ZIP files
- URLs



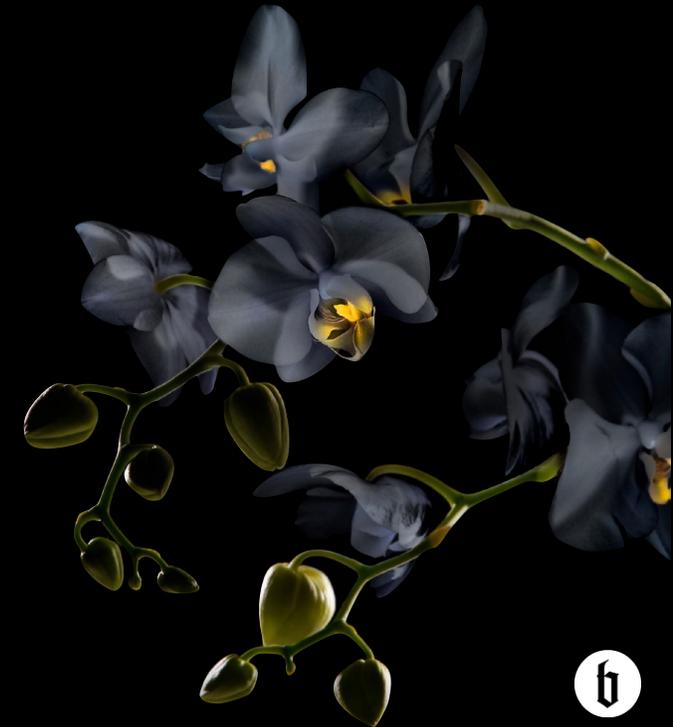
Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders
 - Glob expressions (/folder/*.xml)
 - ZIP files
- URLs
- cURL requests



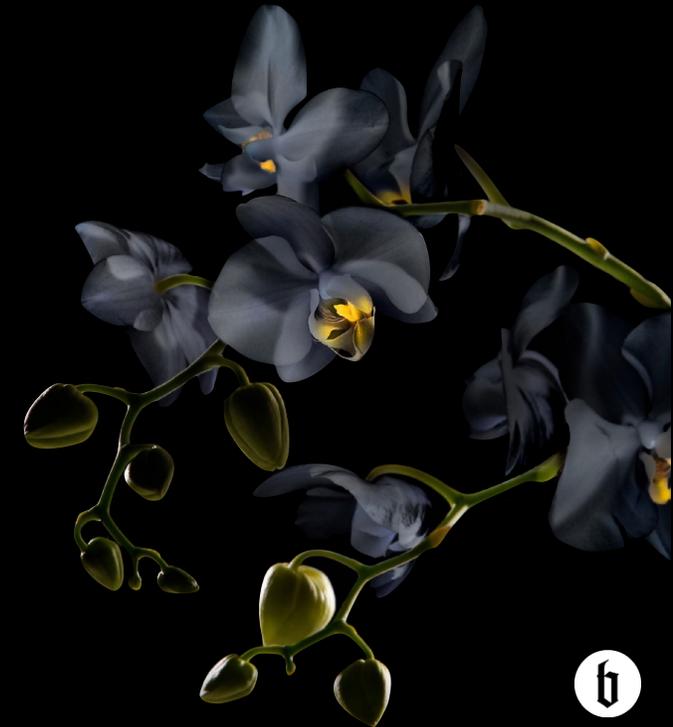
Flexible import resources

- Pimcore assets or filesystem:
 - Files
 - Folders
 - Glob expressions (/folder/*.xml)
 - ZIP files
- URLs
- cURL requests
- Custom PHP scripts



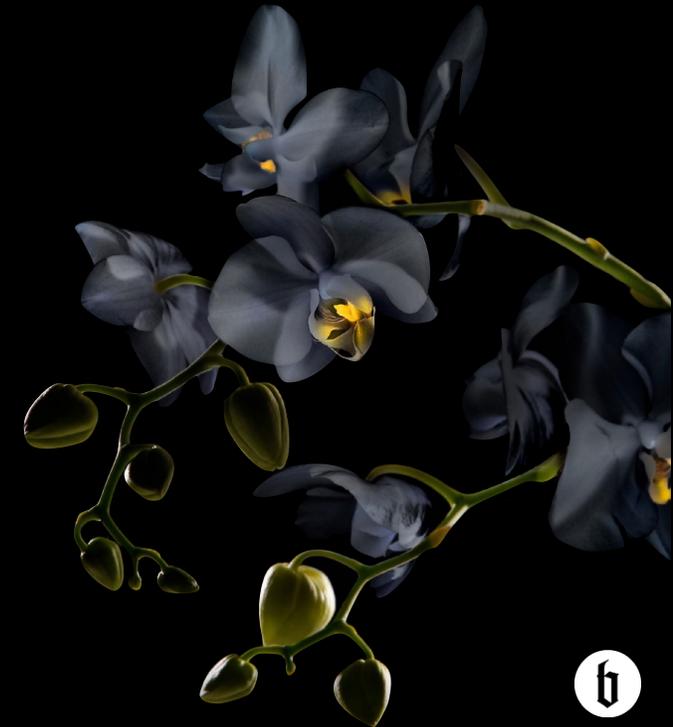
Flexible import source formats

- CSV



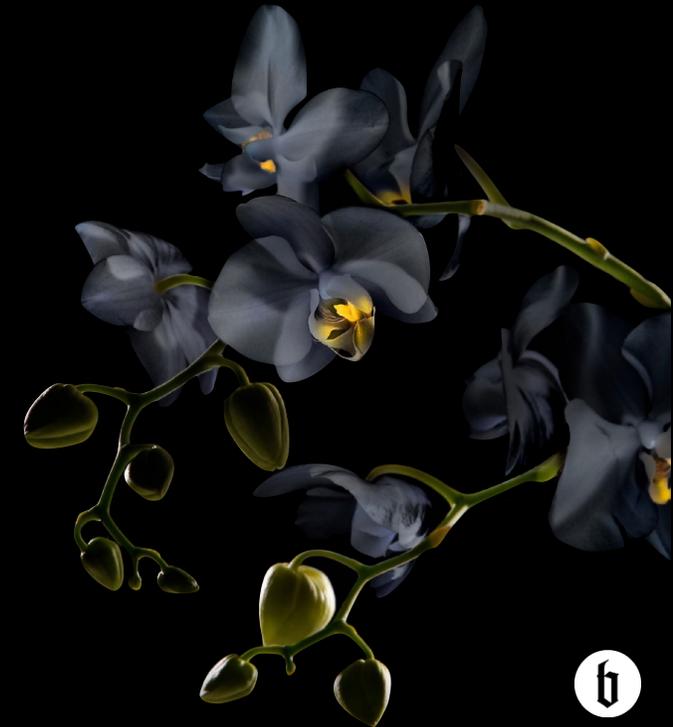
Flexible import source formats

- CSV
- XML / HTML



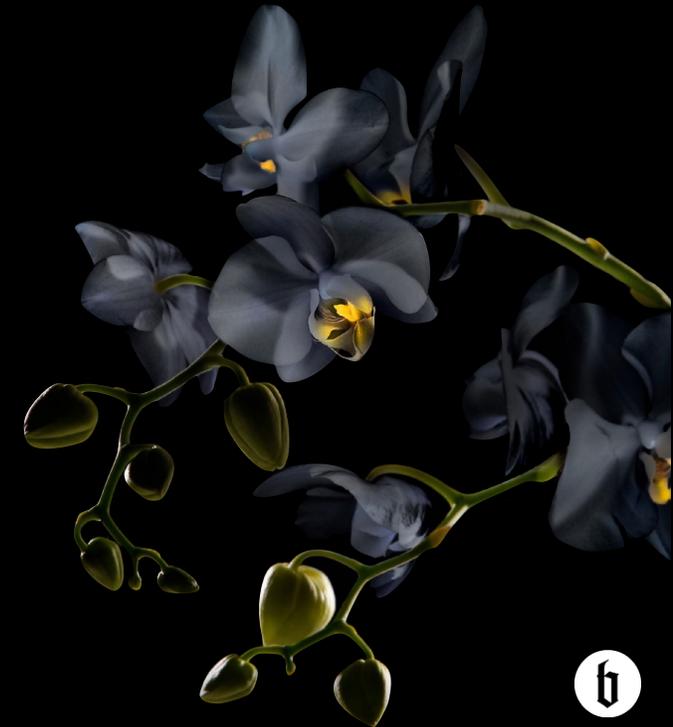
Flexible import source formats

- CSV
- XML / HTML
- JSON



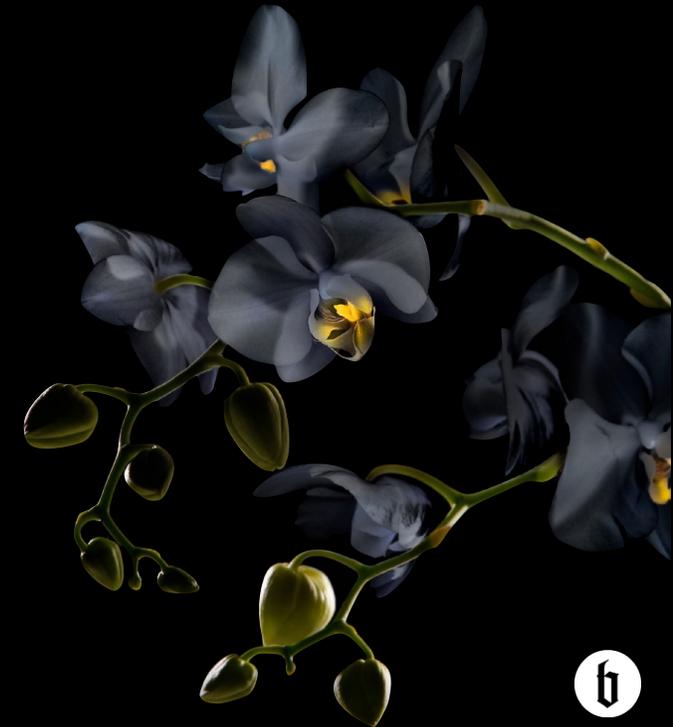
Flexible import source formats

- CSV
- XML / HTML
- JSON
- Excel



Flexible import source formats

- CSV
- XML / HTML
- JSON
- Excel
- Pimcore elements (data objects, assets, documents)



Flexible import source formats

- CSV
- XML / HTML
- JSON
- Excel
- Pimcore elements (data objects, assets, documents)
- Pimcore reports



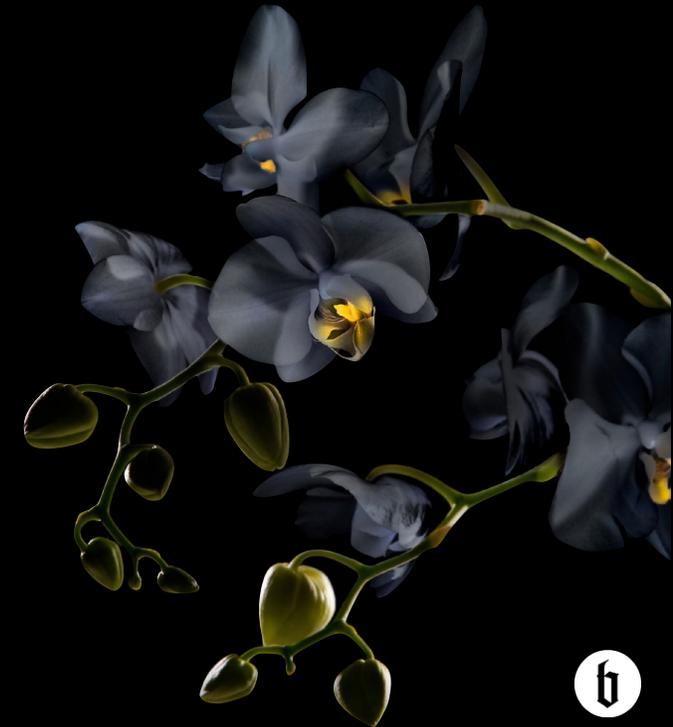
Flexible import source formats

- CSV
- XML / HTML
- JSON
- Excel
- Pimcore elements (data objects, assets, documents)
- Pimcore reports
- File system



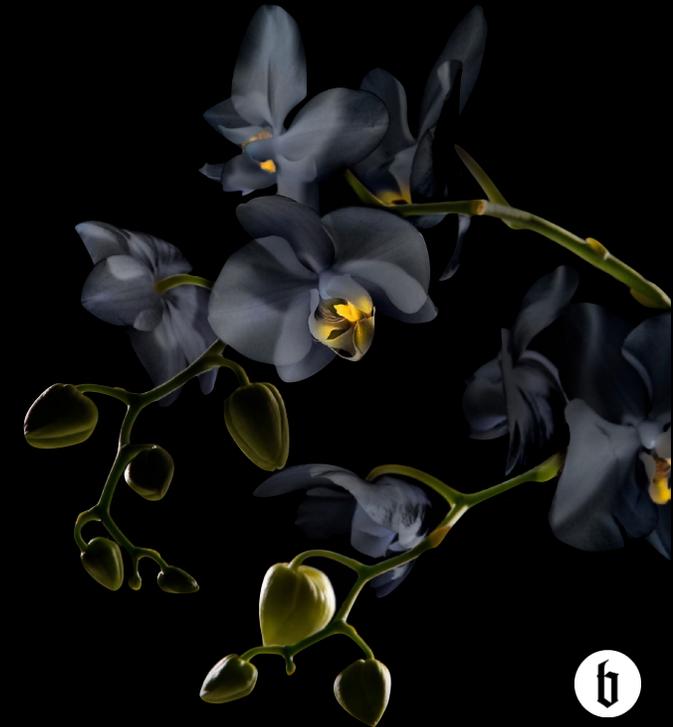
Context-sensitivity

- Skip items if certain criteria are not fulfilled



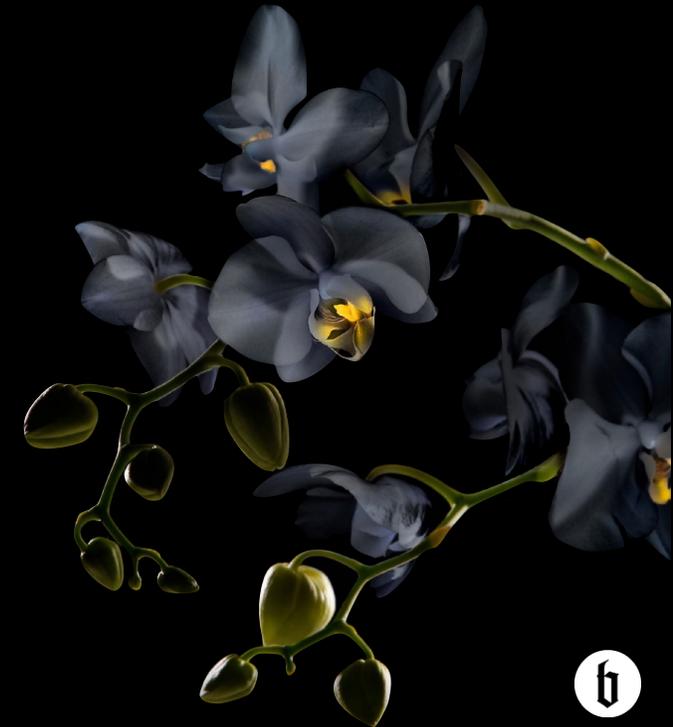
Context-sensitivity

- Skip items if certain criteria are not fulfilled
- Option to not overwrite field value if already filled



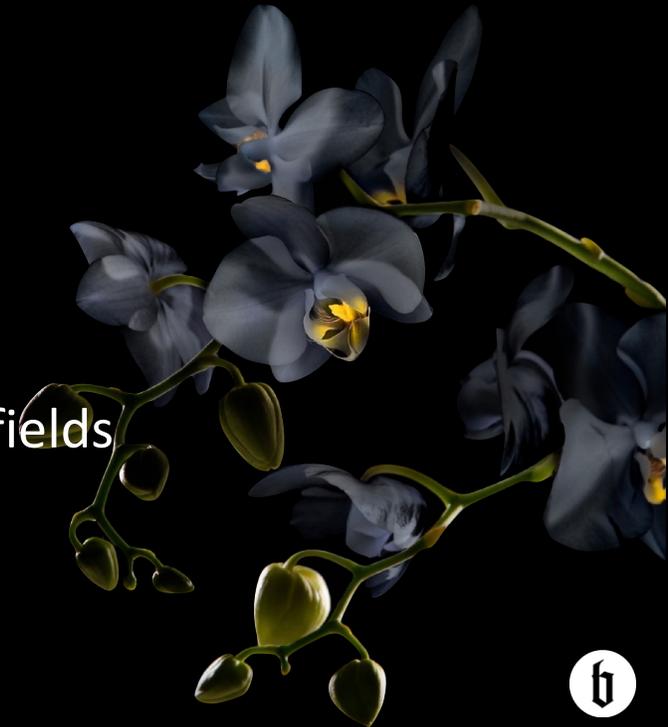
Context-sensitivity

- Skip items if certain criteria are not fulfilled
- Option to not overwrite field value if already filled
- Option to only create new objects / only update existing objects



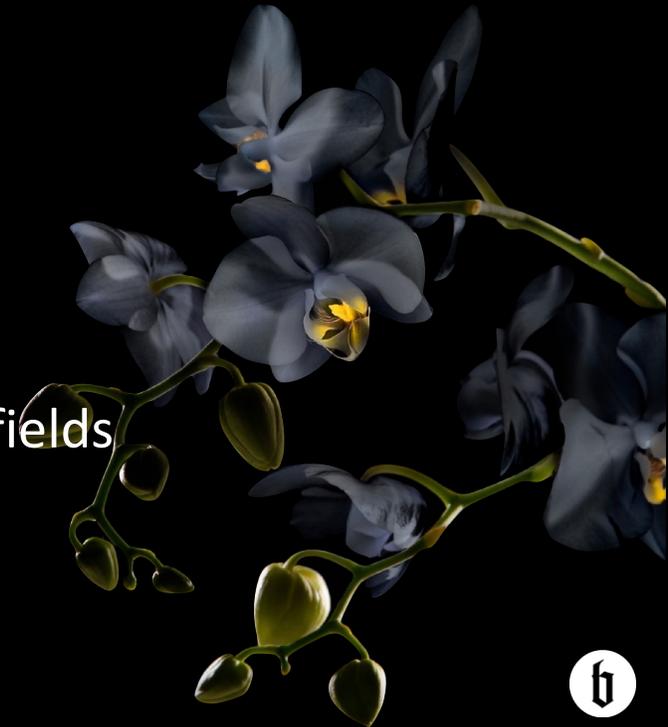
Context-sensitivity

- Skip items if certain criteria are not fulfilled
- Option to not overwrite field value if already filled
- Option to only create new objects / only update existing objects
- Depend import behaviour on value of other import or target object fields



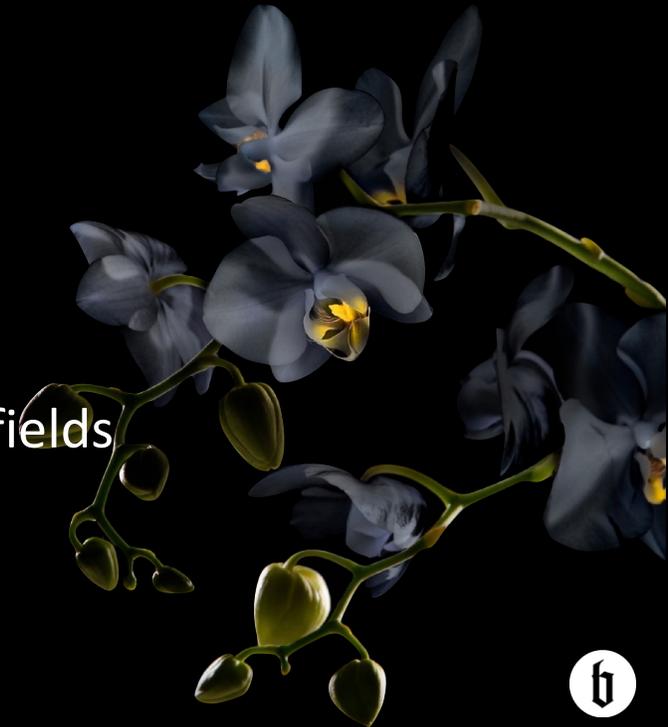
Context-sensitivity

- Skip items if certain criteria are not fulfilled
- Option to not overwrite field value if already filled
- Option to only create new objects / only update existing objects
- Depend import behaviour on value of other import or target object fields
- Create / Update multiple objects from one raw data item



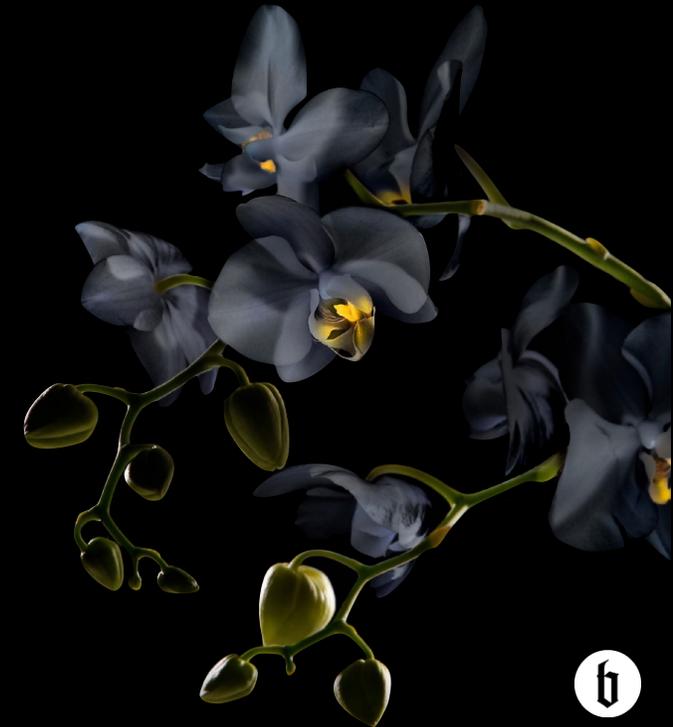
Context-sensitivity

- Skip items if certain criteria are not fulfilled
- Option to not overwrite field value if already filled
- Option to only create new objects / only update existing objects
- Depend import behaviour on value of other import or target object fields
- Create / Update multiple objects from one raw data item
- Correctly handle unpublished versions



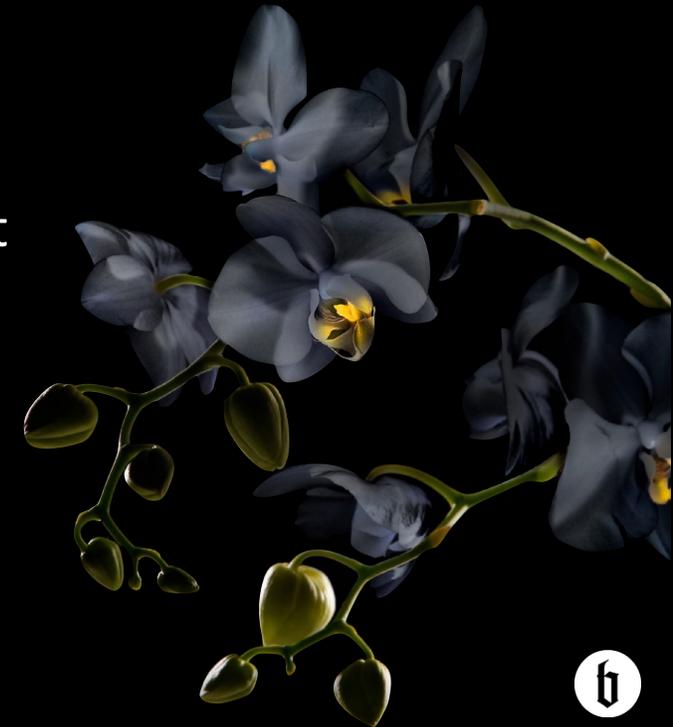
Performance

- Streaming import files → allows for huge import files while keeping memory consumption low



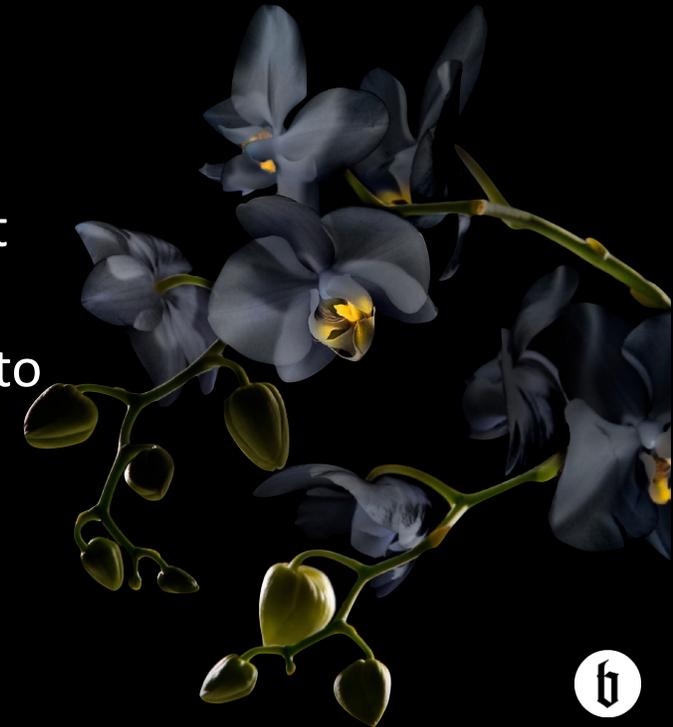
Performance

- Streaming import files → allows for huge import files while keeping memory consumption low
- Option to skip current object if import data did not change since last import



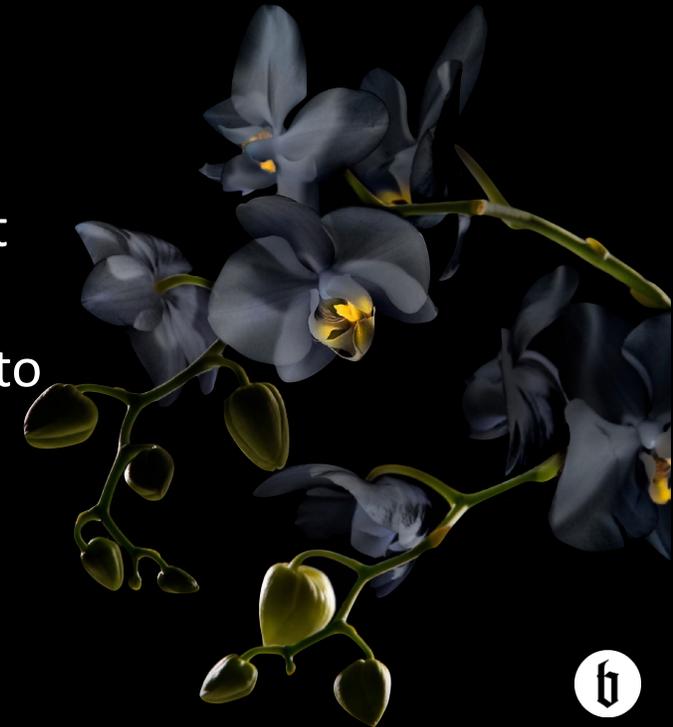
Performance

- Streaming import files → allows for huge import files while keeping memory consumption low
- Option to skip current object if import data did not change since last import
- Fetch related objects only once when multiple raw data items refer to same element



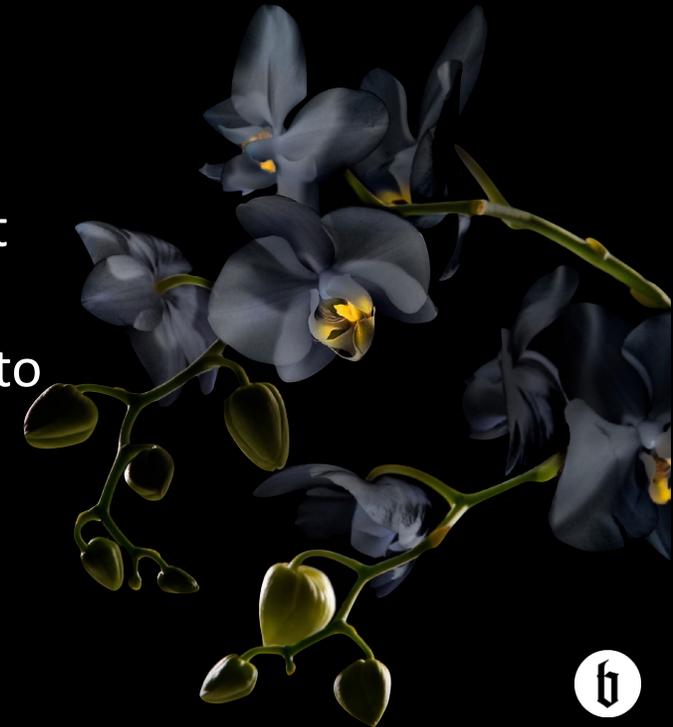
Performance

- Streaming import files → allows for huge import files while keeping memory consumption low
- Option to skip current object if import data did not change since last import
- Fetch related objects only once when multiple raw data items refer to same element
- Dirty Detection for all fields: Save only if import changed any data



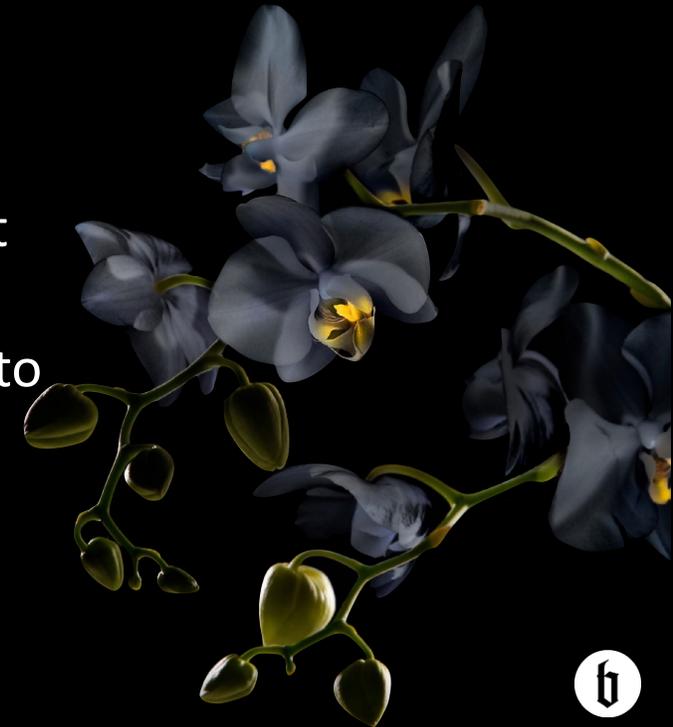
Performance

- Streaming import files → allows for huge import files while keeping memory consumption low
- Option to skip current object if import data did not change since last import
- Fetch related objects only once when multiple raw data items refer to same element
- Dirty Detection for all fields: Save only if import changed any data
- Save only once if multiple raw data items belong to same object

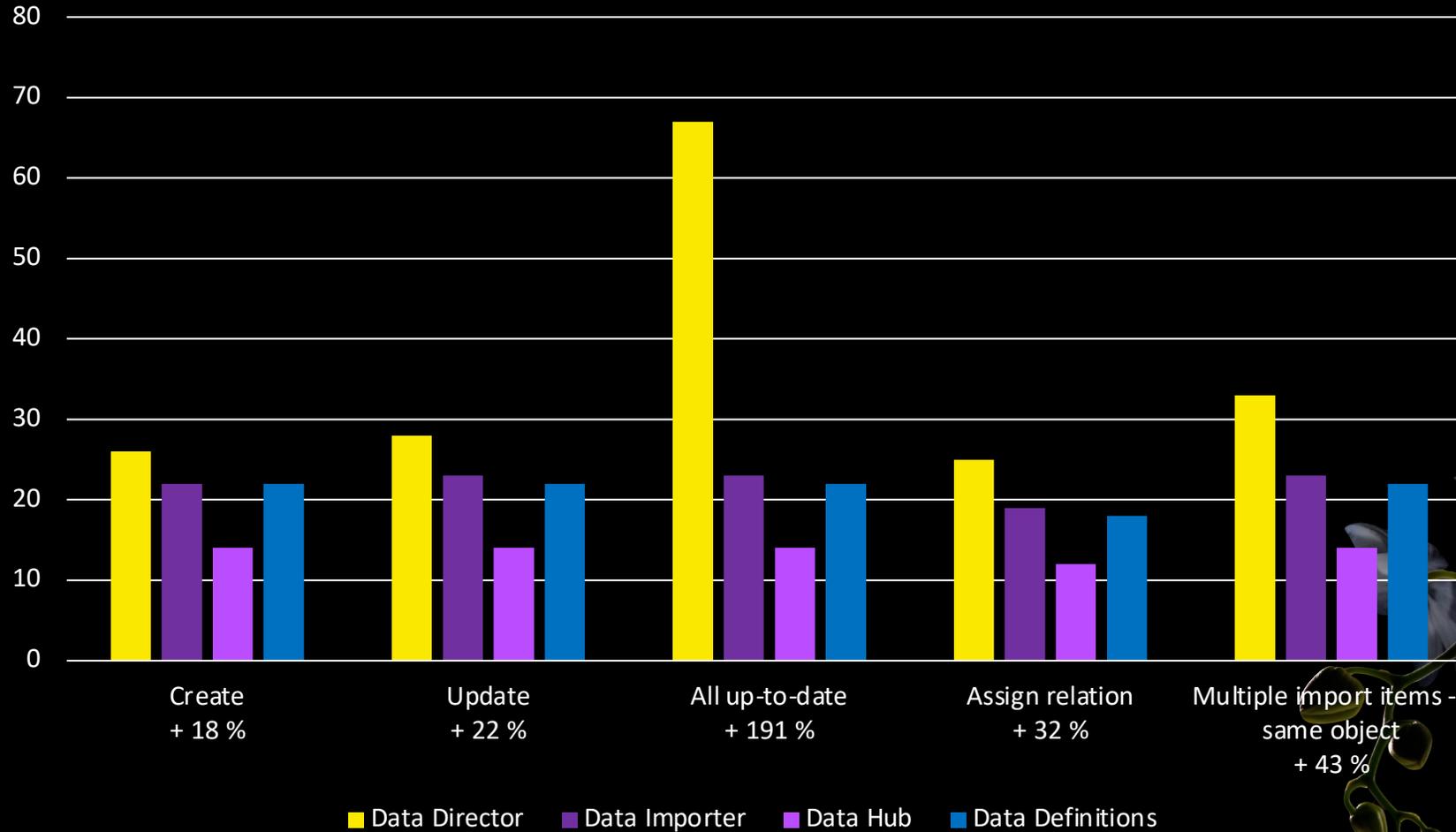


Performance

- Streaming import files → allows for huge import files while keeping memory consumption low
- Option to skip current object if import data did not change since last import
- Fetch related objects only once when multiple raw data items refer to same element
- Dirty Detection for all fields: Save only if import changed any data
- Save only once if multiple raw data items belong to same object
- → Side effect of less saving: cleaner version history



Performance of Pimcore Import bundles Imported objects per second

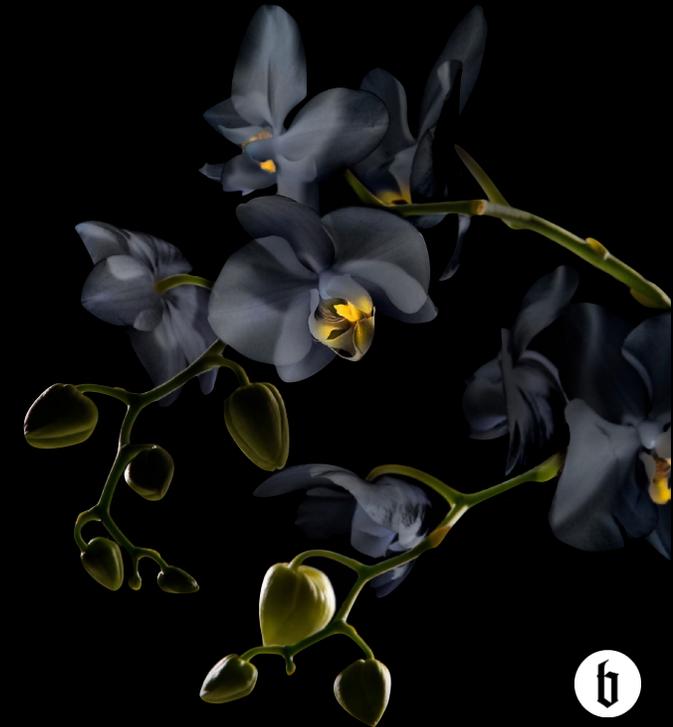


Absolute values will differ depending on complexity of data model + hardware resources

Run automatically on new data:

Automation

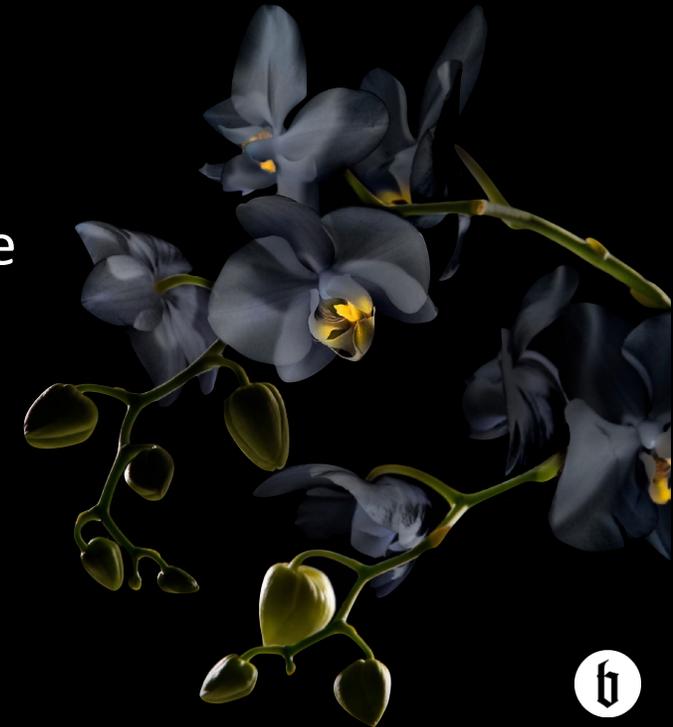
- Automatically start import when a user uploads a CSV file to certain folder



Run automatically on new data:

Automation

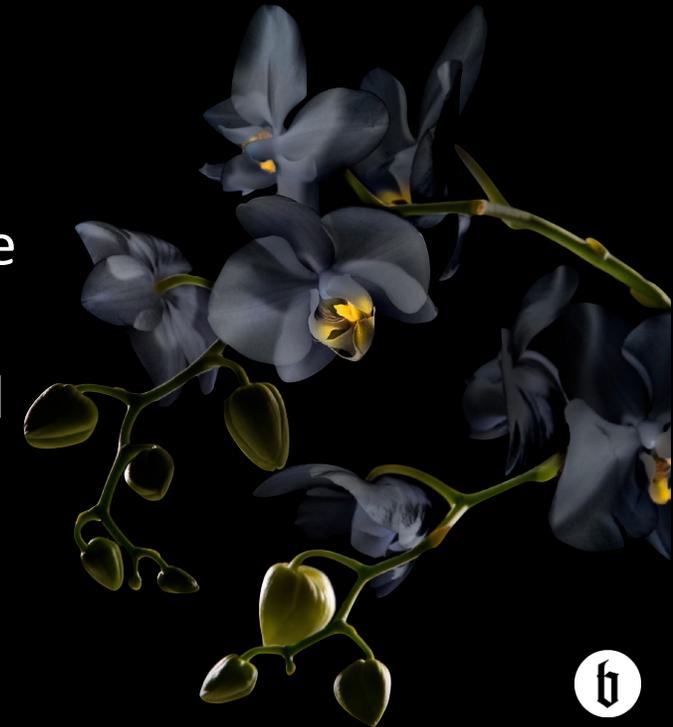
- Automatically start import when a user uploads a CSV file to certain folder
- Trigger assignment of just uploaded asset to data object via file name / metadata



Run automatically on new data:

Automation

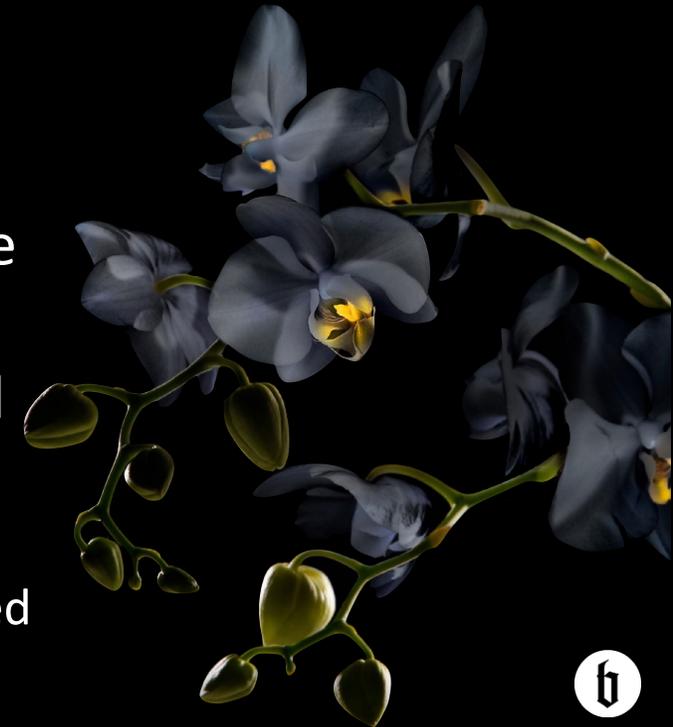
- Automatically start import when a user uploads a CSV file to certain folder
- Trigger assignment of just uploaded asset to data object via file name / metadata
- Run „import“ with same data object as source and target to fill fields based on other fields' values



Run automatically on new data:

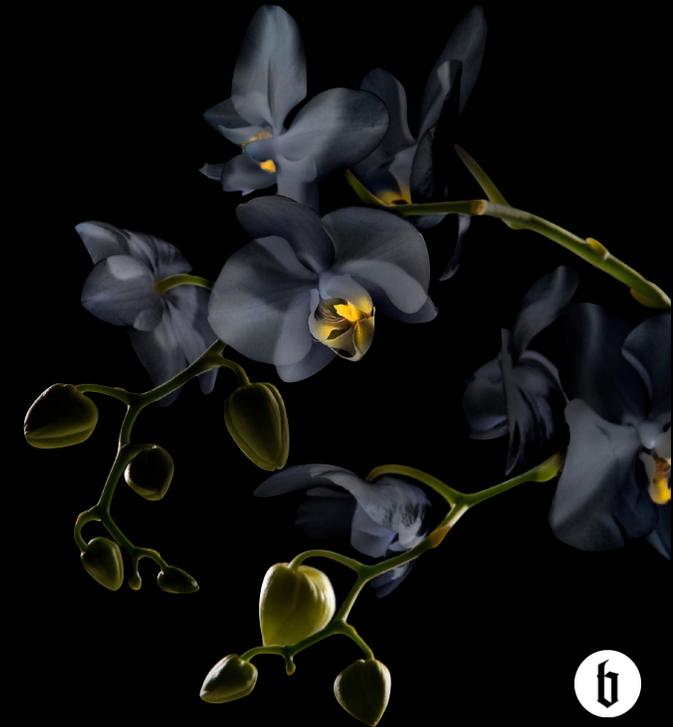
Automation

- Automatically start import when a user uploads a CSV file to certain folder
- Trigger assignment of just uploaded asset to data object via file name / metadata
- Run „import“ with same data object as source and target to fill fields based on other fields' values
 - Same as calculated value field but: calculation can be implemented in Pimcore backend, value can be manually edited, value is only calculated when object gets saved



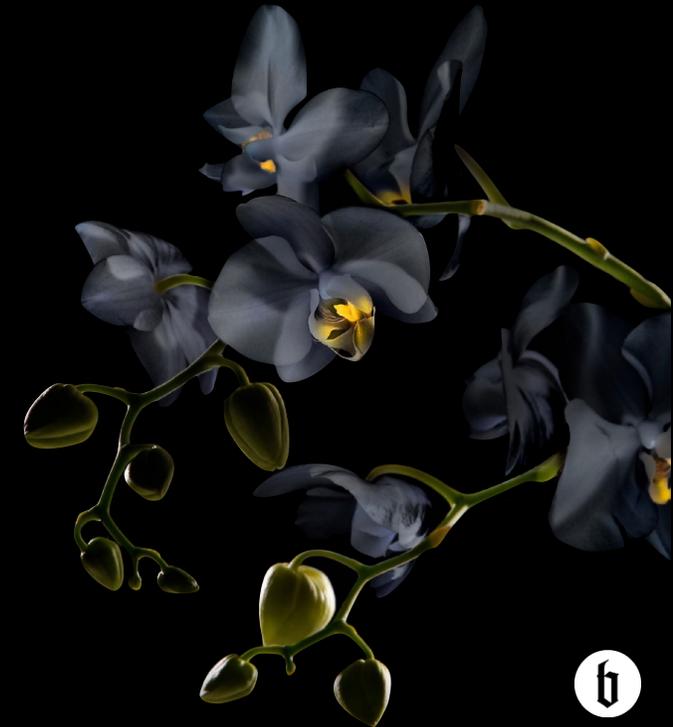
REST API anyone?

- Every dataport can be executed via REST API endpoint



REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Live interfaces between source and target systems



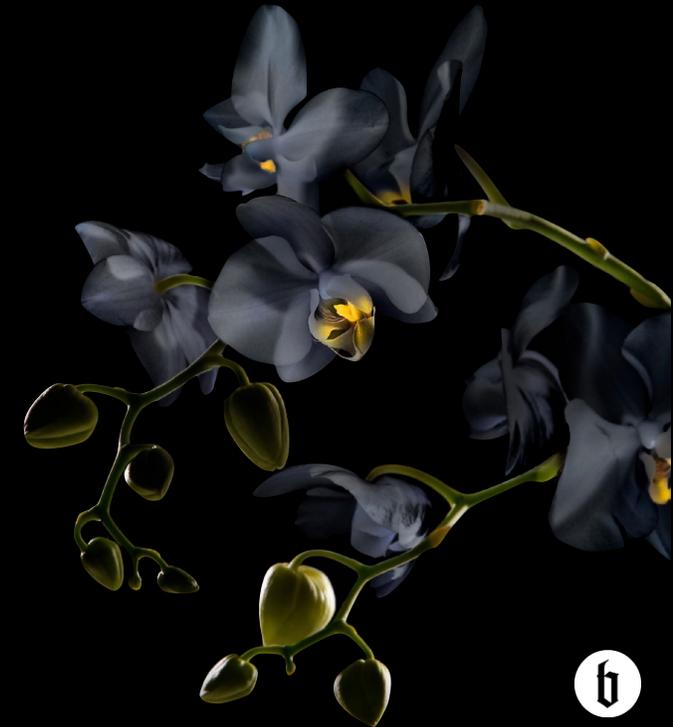
REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Live interfaces between source and target systems
- → No cronjobs, no delays



REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Live interfaces between source and target systems
- → No cronjobs, no delays
- (but of course execution via cronjobs / CLI commands is also possible)



REST API anyone?

REST API Documentation (Dataport "Products from ERP")

POST /api/rest/import/Products+from+ERP Execute import "Products from ERP"

Parameters Try it out

Name	Description
apikey <small>* required</small> string <i>(query)</i>	Rest API key (see 'Permissions & API Keys' button below dataport tree) <i>Default value</i> : c9757b1605bb4da4a83b4a3841b94432

Request body text/csv

Document to be imported (csv). Leave empty to use dataport's default import resource

[Example Value](#) | [Schema](#)

```
"SKU";"Product Name";"Manufacturer";"Categories"
```

Responses

Code	Description	Links
200	Import run successfully. Response document gets returned, if result callback function creates one.	No links
403	Login not possible. Please configure dataport permissions and activate API key for the user which shall execute the import	No links

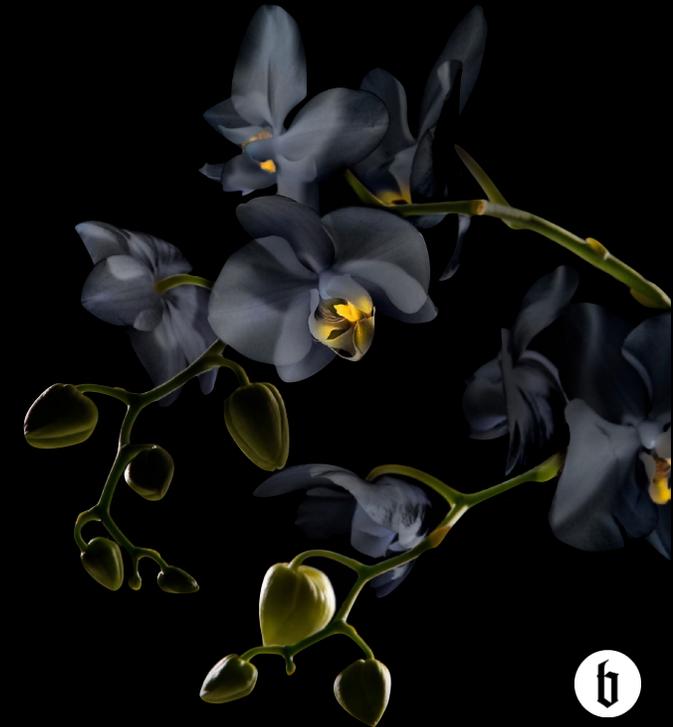


Traceability

- Import archive

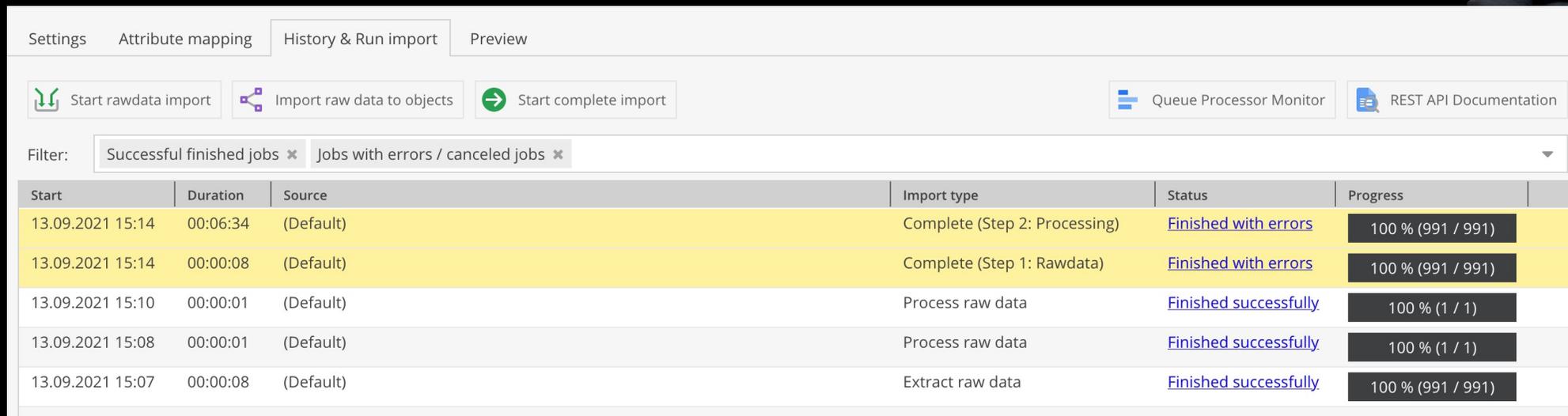
Archive folder:

/import/products-from-erp/archive



Traceability

- Import archive
- Searchable import log history for each import run

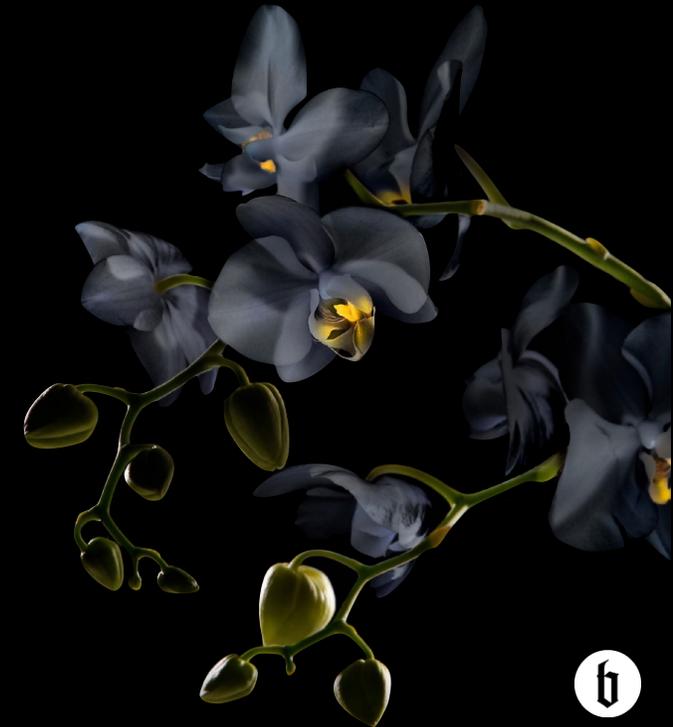


The screenshot displays the 'History & Run import' section of the Blackbit interface. It features a navigation bar with tabs for 'Settings', 'Attribute mapping', 'History & Run import', and 'Preview'. Below the navigation bar are three buttons: 'Start rawdata import', 'Import raw data to objects', and 'Start complete import'. To the right, there are links for 'Queue Processor Monitor' and 'REST API Documentation'. A filter dropdown is set to 'Successful finished jobs', with 'Jobs with errors / canceled jobs' also visible. The main content is a table with the following data:

Start	Duration	Source	Import type	Status	Progress
13.09.2021 15:14	00:06:34	(Default)	Complete (Step 2: Processing)	Finished with errors	100 % (991 / 991)
13.09.2021 15:14	00:00:08	(Default)	Complete (Step 1: Rawdata)	Finished with errors	100 % (991 / 991)
13.09.2021 15:10	00:00:01	(Default)	Process raw data	Finished successfully	100 % (1 / 1)
13.09.2021 15:08	00:00:01	(Default)	Process raw data	Finished successfully	100 % (1 / 1)
13.09.2021 15:07	00:00:08	(Default)	Extract raw data	Finished successfully	100 % (991 / 991)

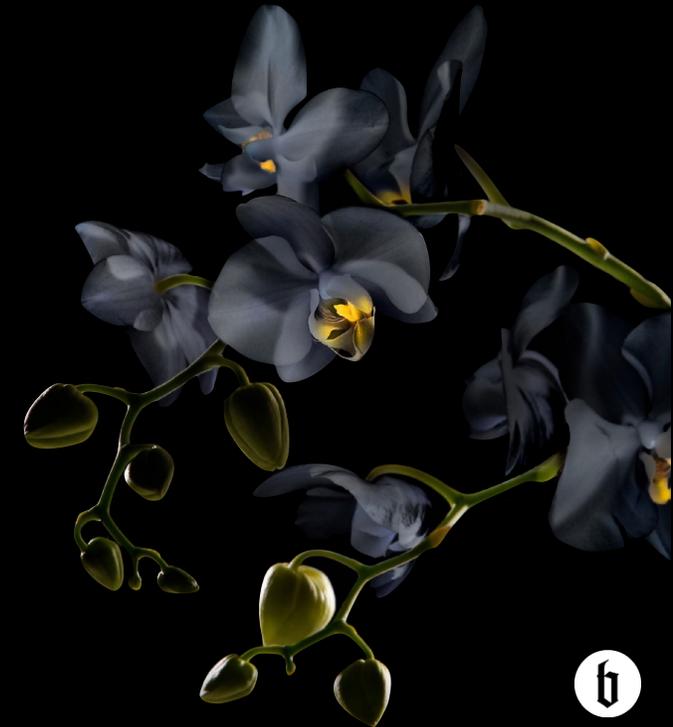
Traceability

- Import archive
- Searchable import log history for each import run
- Automatic email notification if **ANYTHING** unusual happened during import



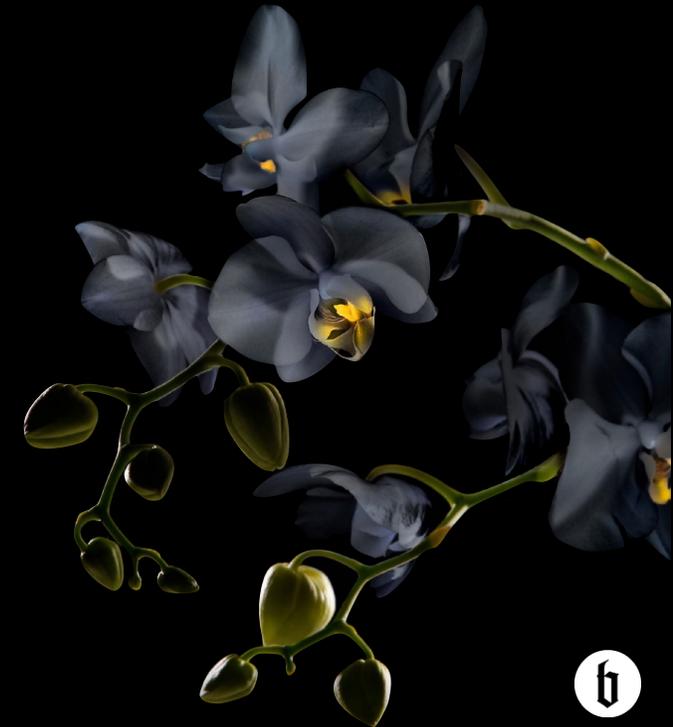
Traceability

- Import archive
- Searchable import log history for each import run
- Automatic email notification if **ANYTHING** unusual happened during import
- Option to revert import fields to certain point in time



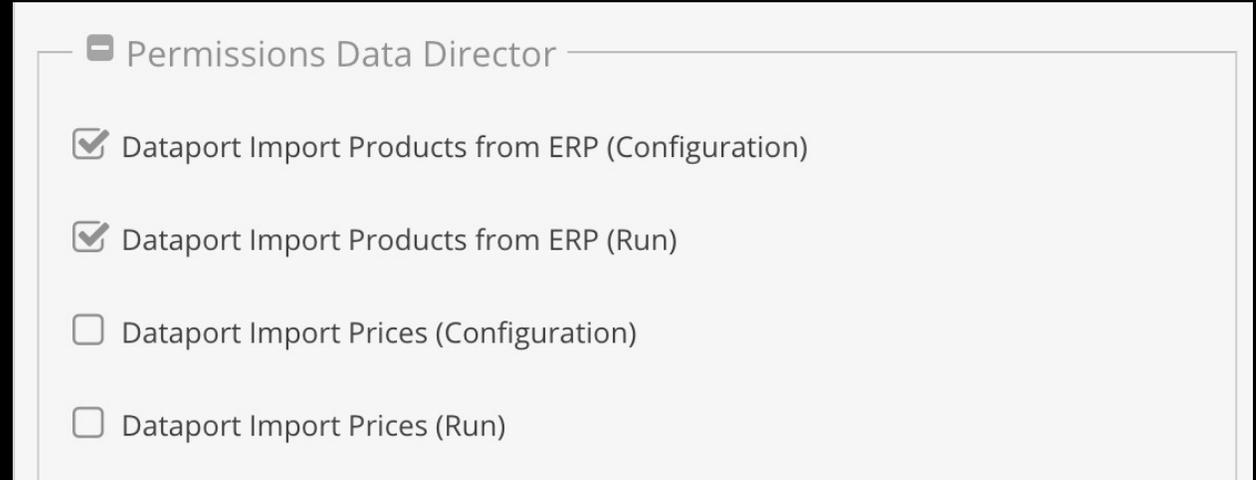
Traceability

- Import archive
- Searchable import log history for each import run
- Automatic email notification if ANYTHING unusual happened during import
- Option to revert import fields to certain point in time
- In development: versioning for import configuration



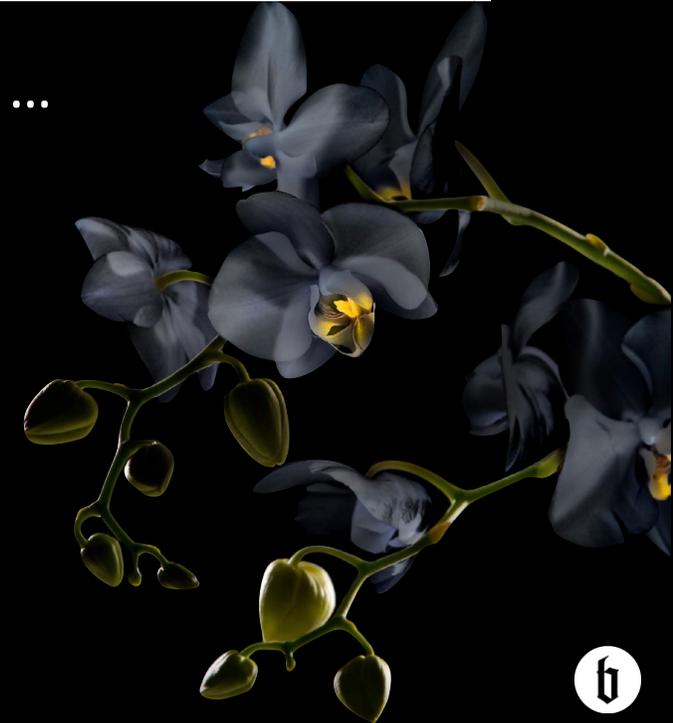
Permissions

- For each dataport it is configurable which users are allowed to ...
 - Run dataport
 - Configure dataport



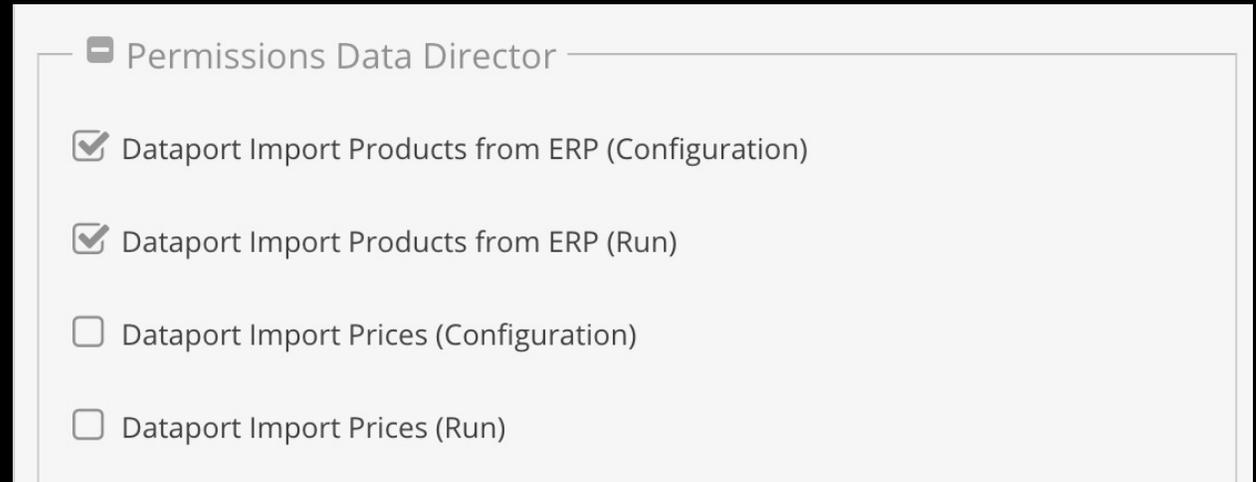
The screenshot shows a configuration window titled 'Permissions Data Director'. It contains a list of four items, each with a checkbox:

- Dataport Import Products from ERP (Configuration)
- Dataport Import Products from ERP (Run)
- Dataport Import Prices (Configuration)
- Dataport Import Prices (Run)



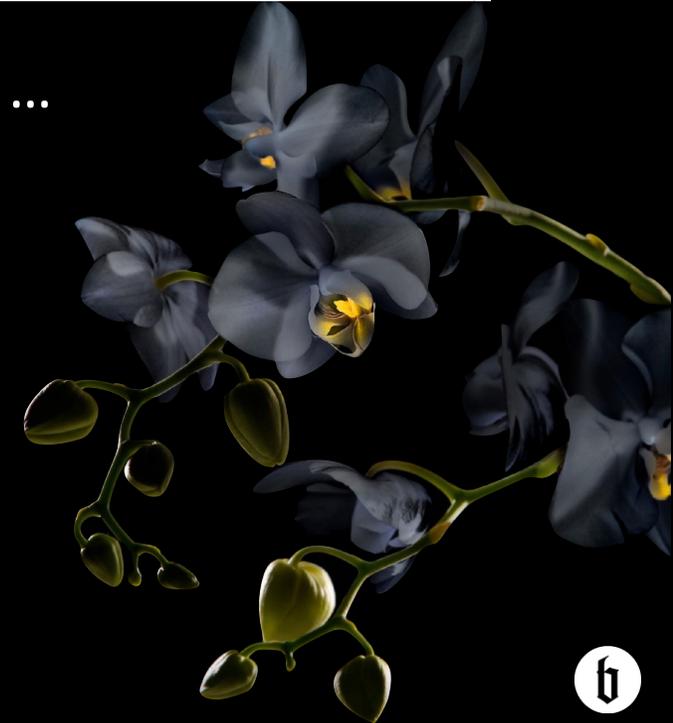
Permissions

- For each dataport it is configurable which users are allowed to ...
 - Run dataport
 - Configure dataport
- Pimcore's standard role-based user permissions are used



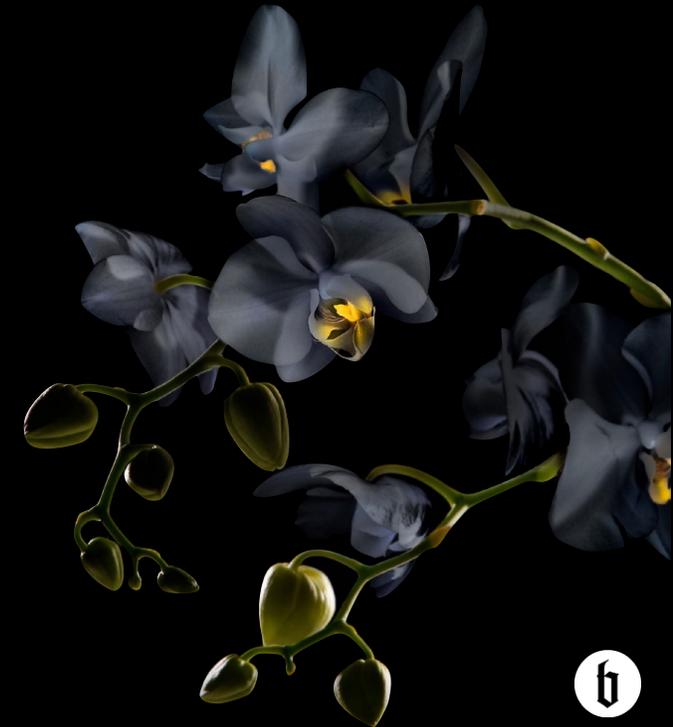
The screenshot shows a configuration window titled 'Permissions Data Director'. It contains a list of four items, each with a checkbox:

- Dataport Import Products from ERP (Configuration)
- Dataport Import Products from ERP (Run)
- Dataport Import Prices (Configuration)
- Dataport Import Prices (Run)



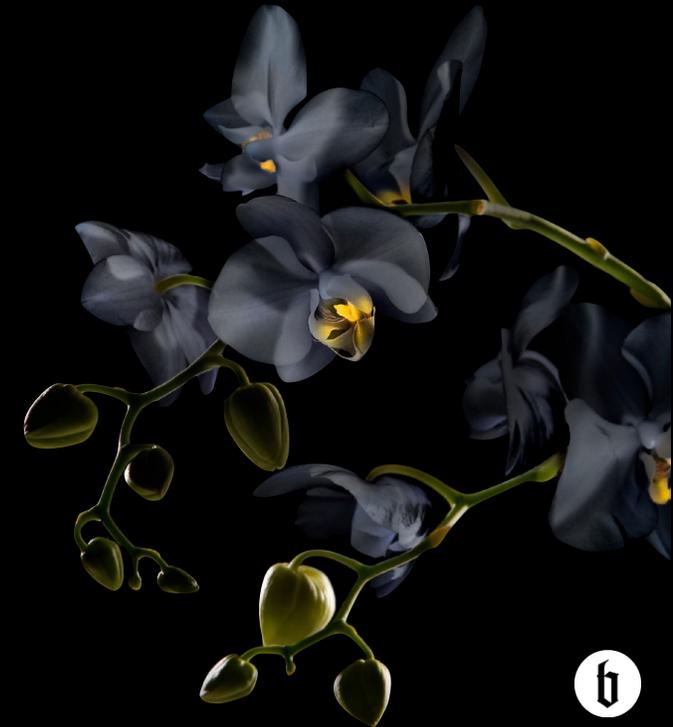
Dependent imports / Import pipelines

- Chain multiple imports to be executed one after another



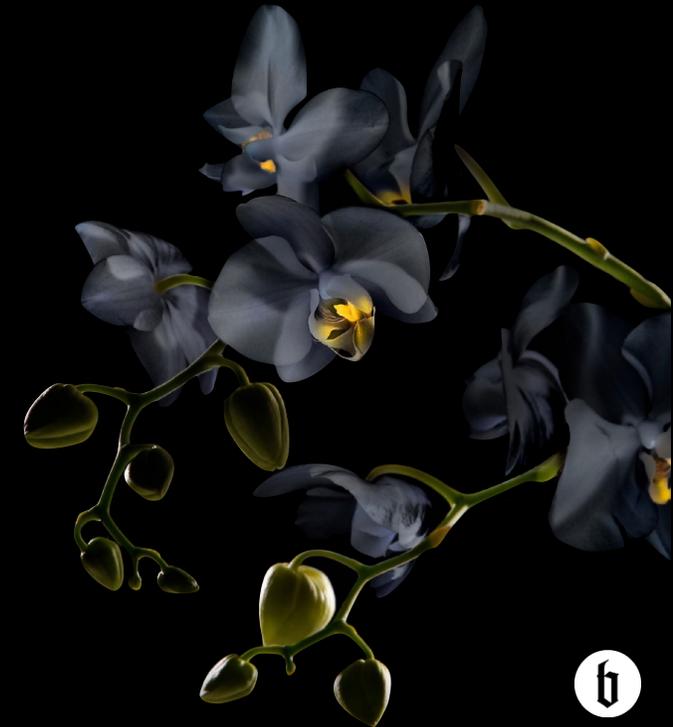
Dependent imports / Import pipelines

- Chain multiple imports to be executed one after another
- Call another import with data from the first import as import source for the follow-up



Dependent imports / Import pipelines

- Chain multiple imports to be executed one after another
- Call another import with data from the first import as import source for the follow-up
- Check if relation object already exists -> if not, call other **parametrized** import to create it and then continue with first import

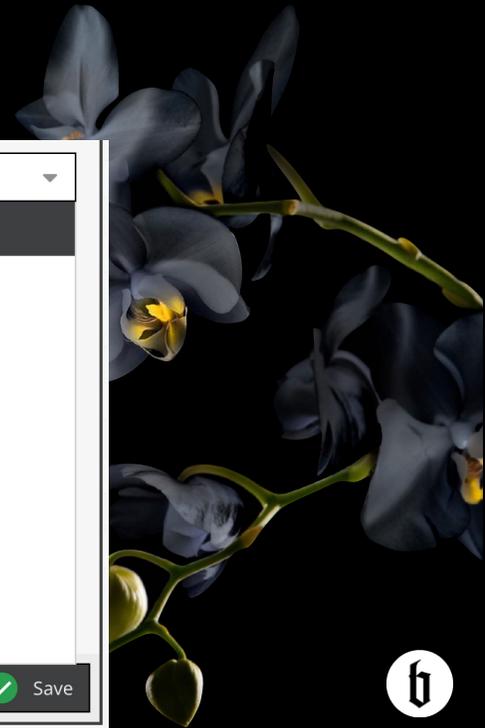


And the really cool things?

- Automatic translation with DeepL / AWS Translate



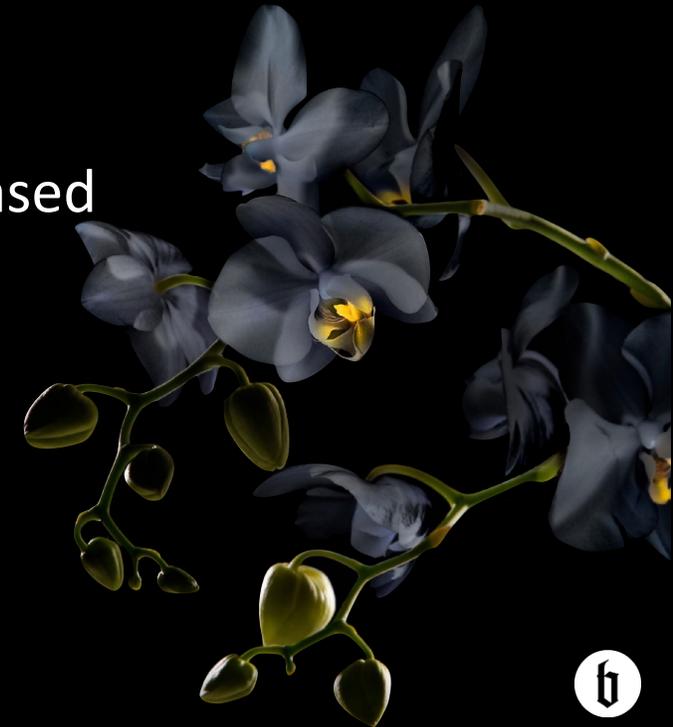
The screenshot shows a user interface for automatic translation. On the left, there is a table with a single row containing the number '1'. To the right of the table is a dropdown menu labeled 'Auto-translate from:'. The dropdown is currently set to 'Disabled' and is open, showing a list of languages with their respective flags: Dutch, English, French, German, Italian, Polish, Portuguese, and Russian. At the bottom right of the interface, there is a 'Save' button with a green checkmark icon.



And the really cool things?

- Automatic translation with DeepL / AWS Translate
- Artificial Intelligence, e.g. to automatically assign categories based on product name and description

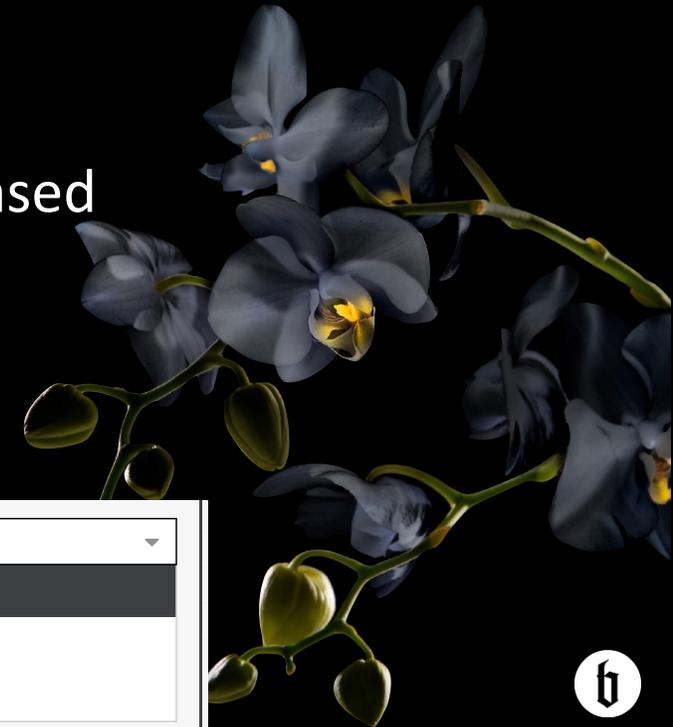
Automatically assign:



And the really cool things?

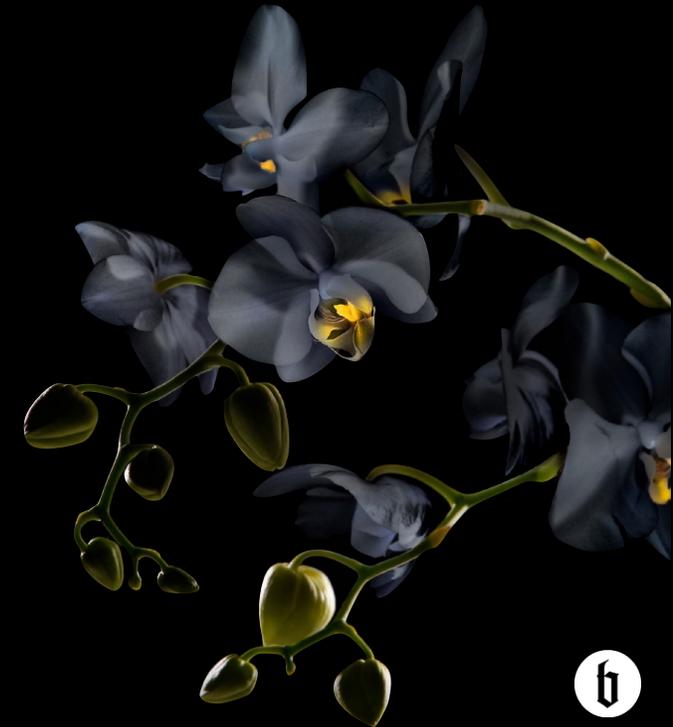
- Automatic translation with DeepL / AWS Translate
- Artificial Intelligence, e.g. to automatically assign categories based on product name and description
- Optimization: Minimize / Maximize calculated value field by changing certain data fields

Optimize:	Disabled
Template:	Disabled
1	Minimize
	Maximize



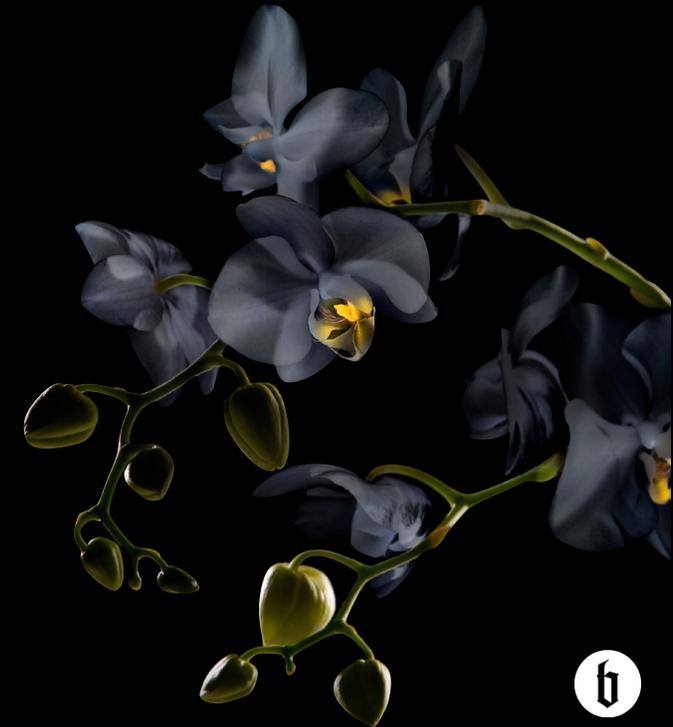
And the really cool things?

- Every mentioned feature is possible without programming

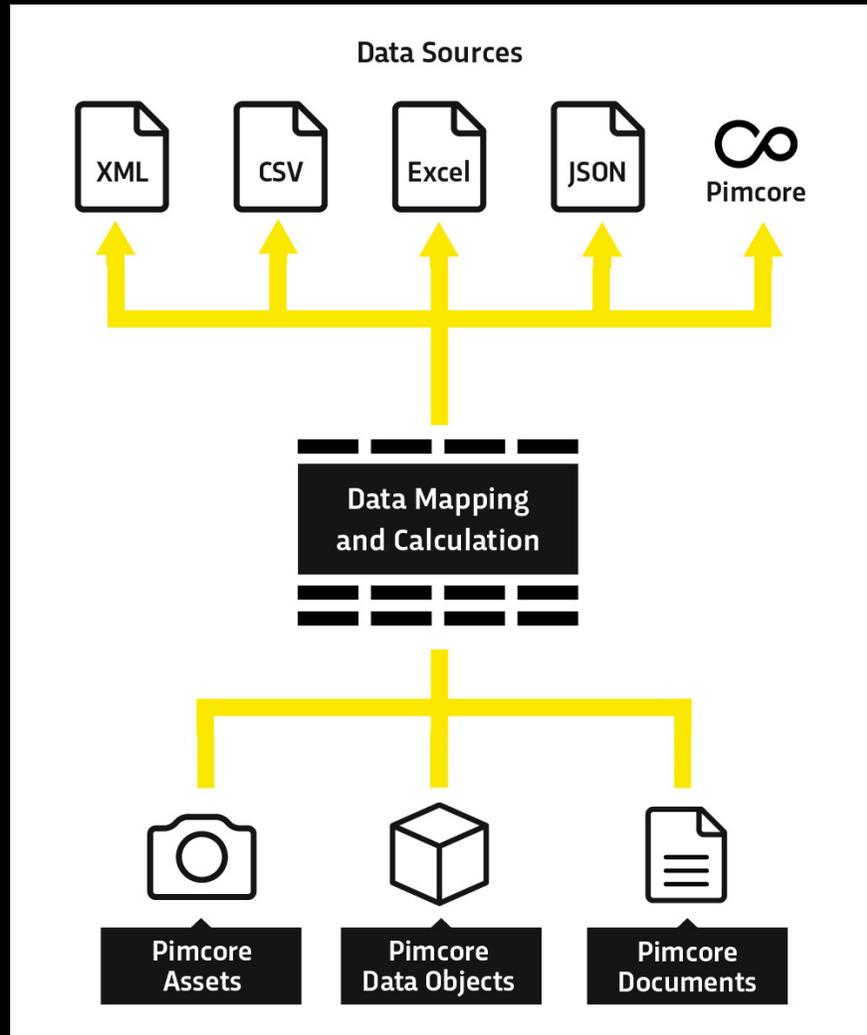


And the really cool things?

- Every mentioned feature is possible without programming
- But you can always add own logic via callback functions



Live Demo Export



blackbit

digital Commerce

There are plenty of other fish in the sea

Advantages of Data Director compared to other bundles

Exports



<>	Result callback function	<>
<>	Result document action	<>

„But my target system needs a very complex XML format“

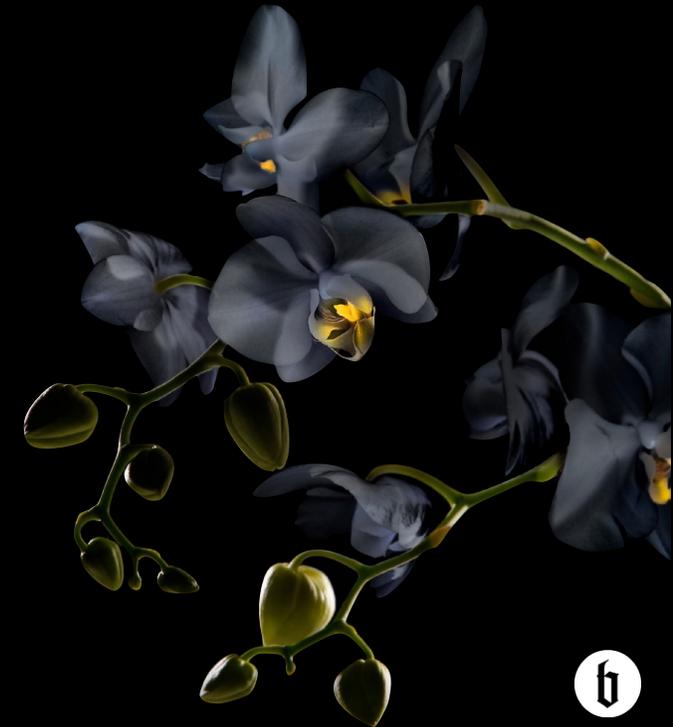
- Export format is fully flexible



<>	Result callback function	<>
<>	Result document action	<>

„But my target system needs a very complex XML format“

- Export format is fully flexible
- Common use-case export formats included:



<>	Result callback function	<>
<>	Result document action	<>

„But my target system needs a very complex XML format“

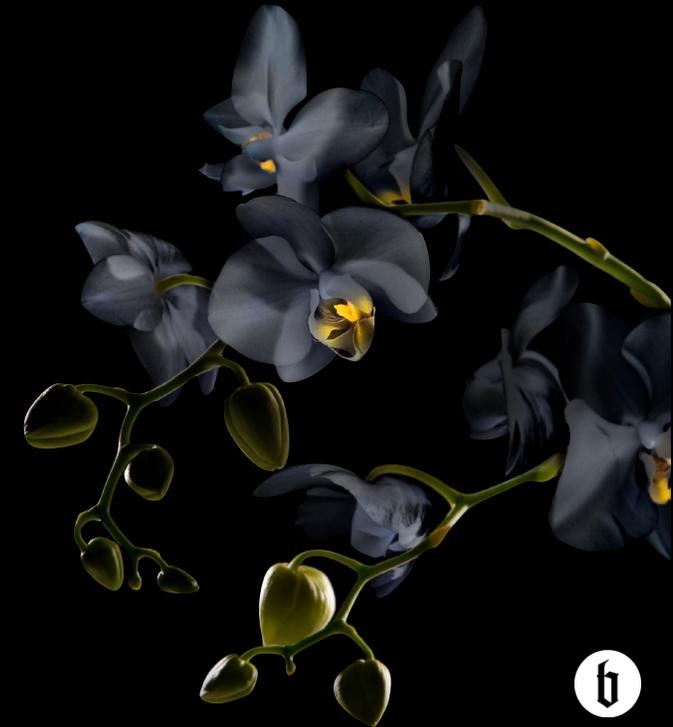
- Export format is fully flexible
- Common use-case export formats included:
 - CSV, JSON, XML, Excel





„But my target system needs a very complex XML format“

- Export format is fully flexible
- Common use-case export formats included:
 - CSV, JSON, XML, Excel
 - All above including referenced assets, compressed to ZIP





„But my target system needs a very complex XML format“

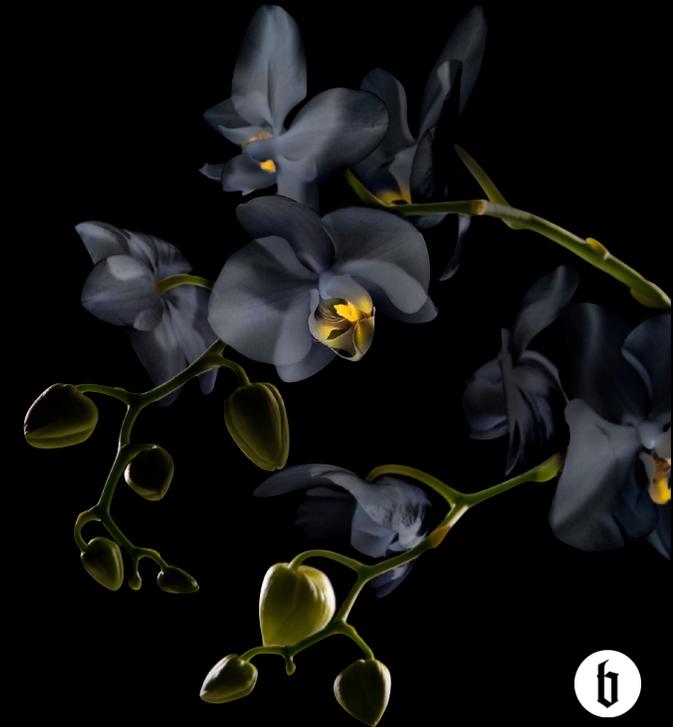
- Export format is fully flexible
- Common use-case export formats included:
 - CSV, JSON, XML, Excel
 - All above including referenced assets, compressed to ZIP
 - Google Shopping, Facebook Products, BMEcat available as add-on bundles





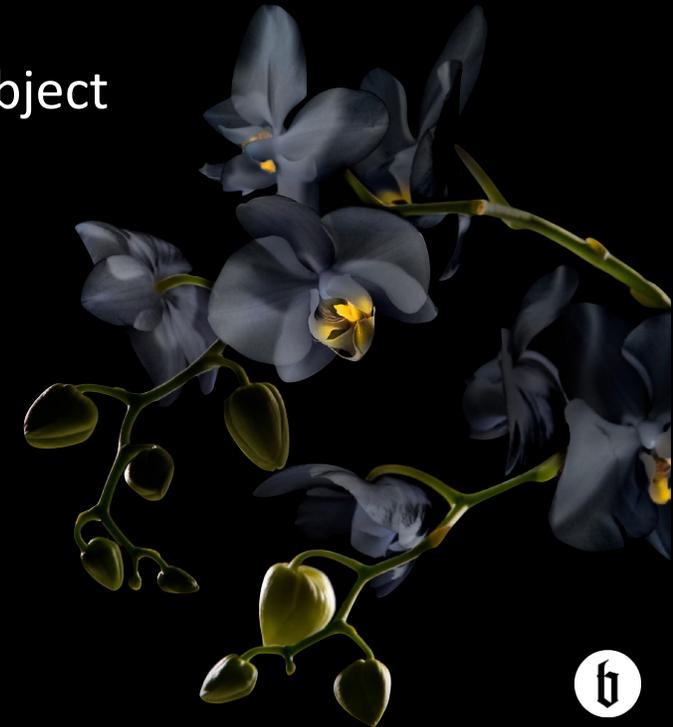
„But my target system needs a very complex XML format“

- Export format is fully flexible
- Common use-case export formats included:
 - CSV, JSON, XML, Excel
 - All above including referenced assets, compressed to ZIP
 - Google Shopping, Facebook Products, BMEcat available as add-on bundles
- Exports to APIs available as add-on bundles
 - xt:commerce, Hubspot, JTL
 - In development: Shopify, Shopware, Prestashop, BigCommerce



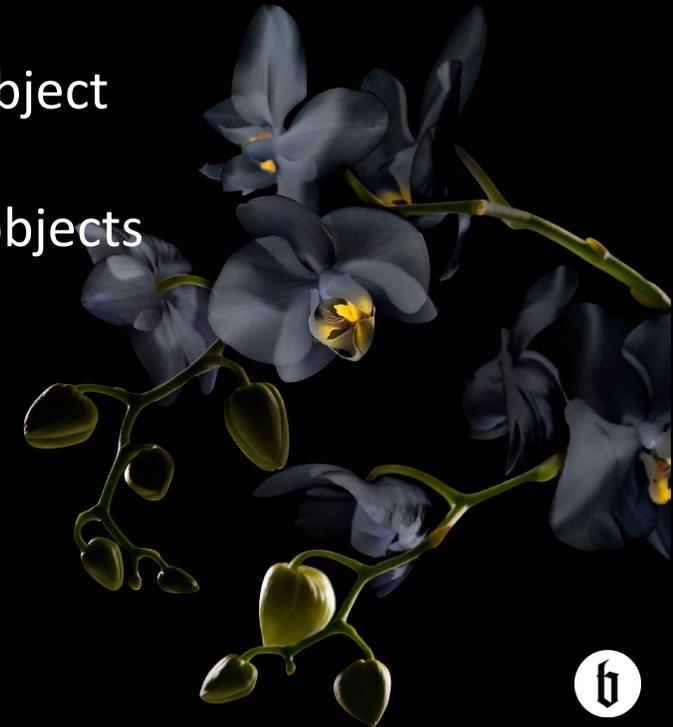
Semper Paratus – Always ready

- Export data gets extracted / updated in background process when object of source data class gets saved



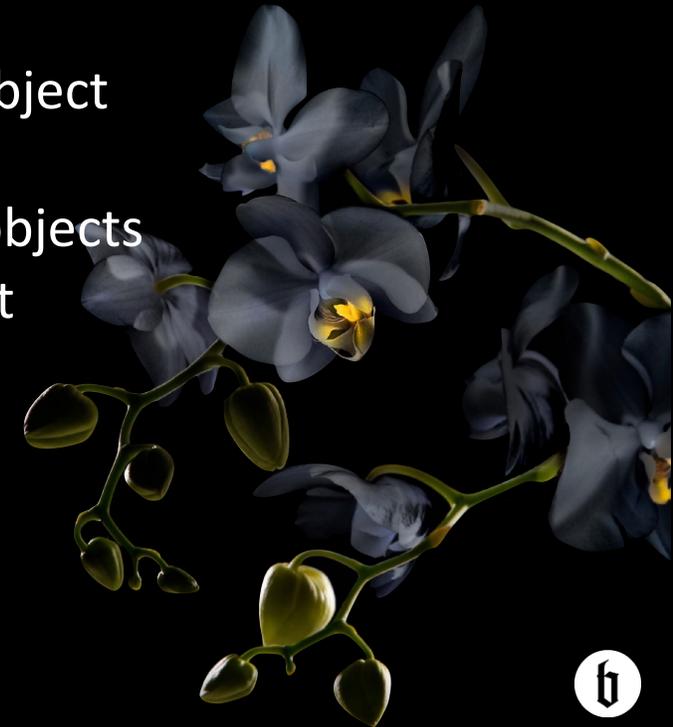
Semper Paratus – Always ready

- Export data gets extracted / updated in background process when object of source data class gets saved
- when export is requested, no data has to be fetched from Pimcore objects



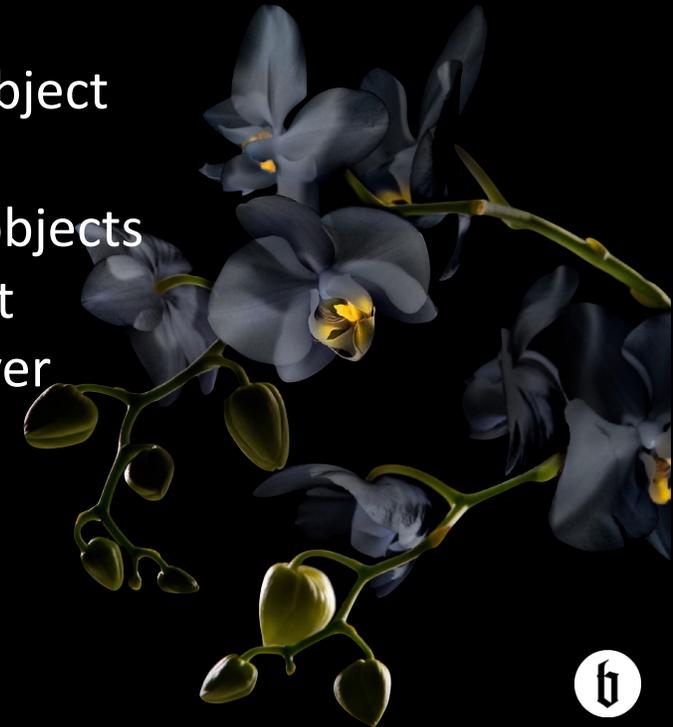
Semper Paratus – Always ready

- Export data gets extracted / updated in background process when object of source data class gets saved
- when export is requested, no data has to be fetched from Pimcore objects
- only export document has to be generated with the data → very fast



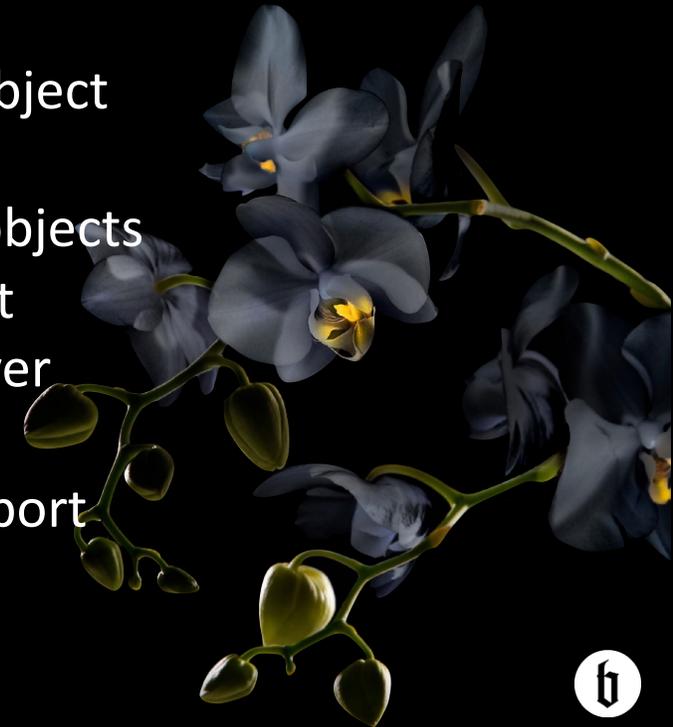
Semper Paratus – Always ready

- Export data gets extracted / updated in background process when object of source data class gets saved
- when export is requested, no data has to be fetched from Pimcore objects
- only export document has to be generated with the data → very fast
- Intelligent check if anything changed since last export → if not, deliver cached export document → even faster



Semper Paratus – Always ready

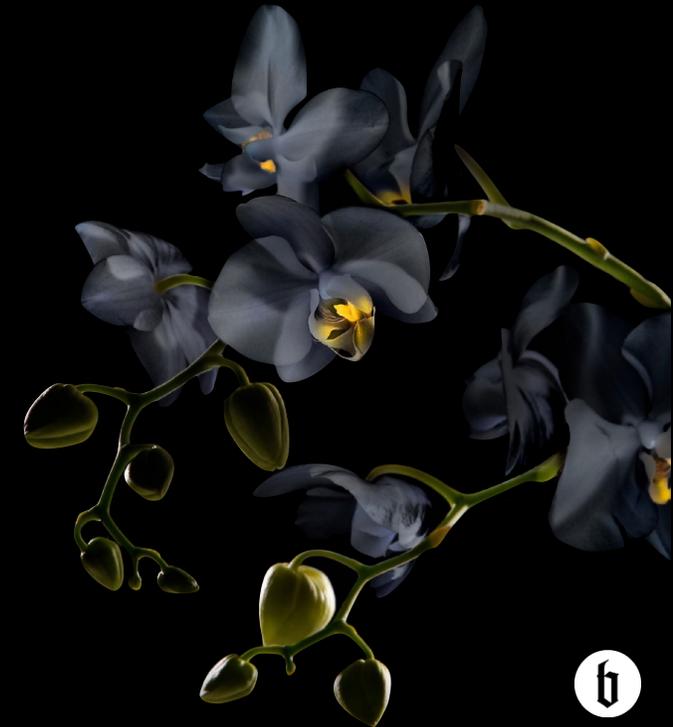
- Export data gets extracted / updated in background process when object of source data class gets saved
- when export is requested, no data has to be fetched from Pimcore objects
- only export document has to be generated with the data → very fast
- Intelligent check if anything changed since last export → if not, deliver cached export document → even faster
- And if nothing changed and the client already has the up-to-date export document in cache, client does not even need to download export document again



Run automatically on new data:

Automatic exports

- Trigger export to target system when an object changes



Run automatically on new data:

Automatic exports

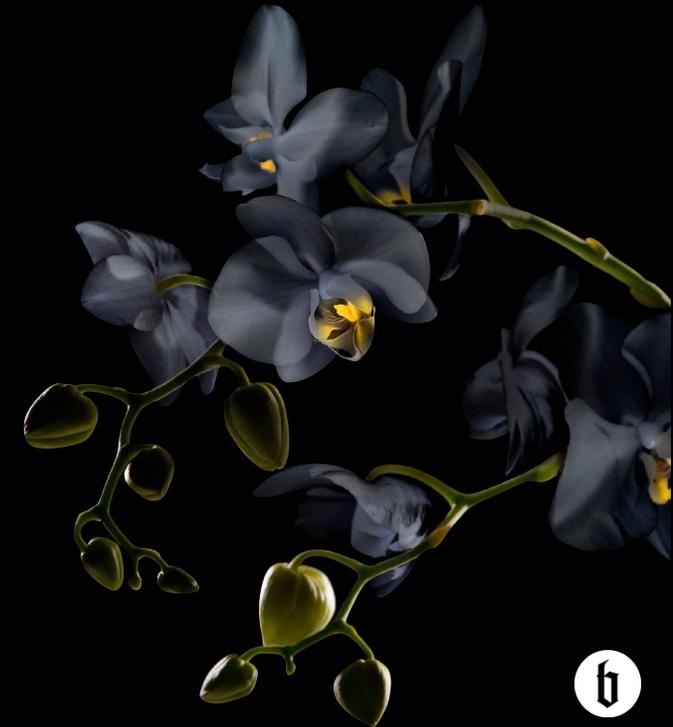
- Trigger export to target system when an object changes
→ e.g. Push API to webshop / ERP system



Run automatically on new data:

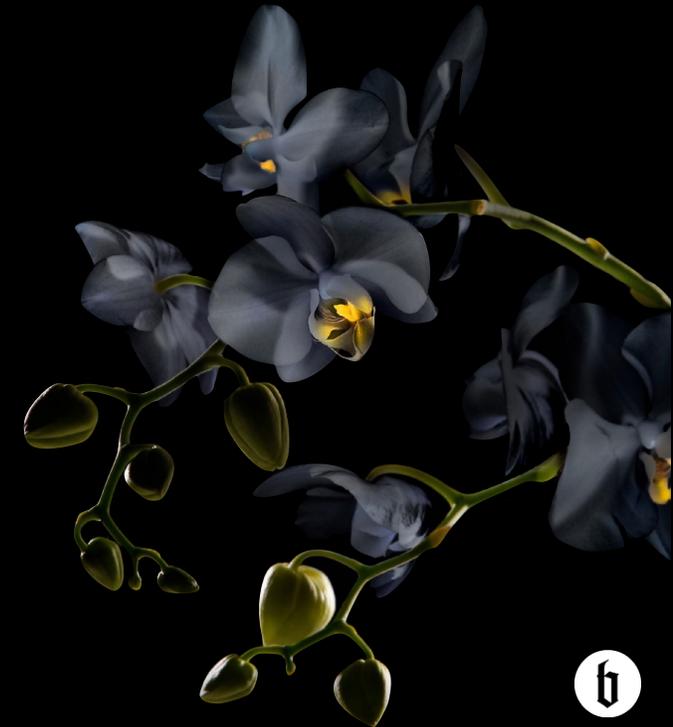
Automatic exports

- Trigger export to target system when an object changes
→ e.g. Push API to webshop / ERP system
- When object of dataport's source class gets saved, generate export document (e.g. product datasheet) and upload it via FTP



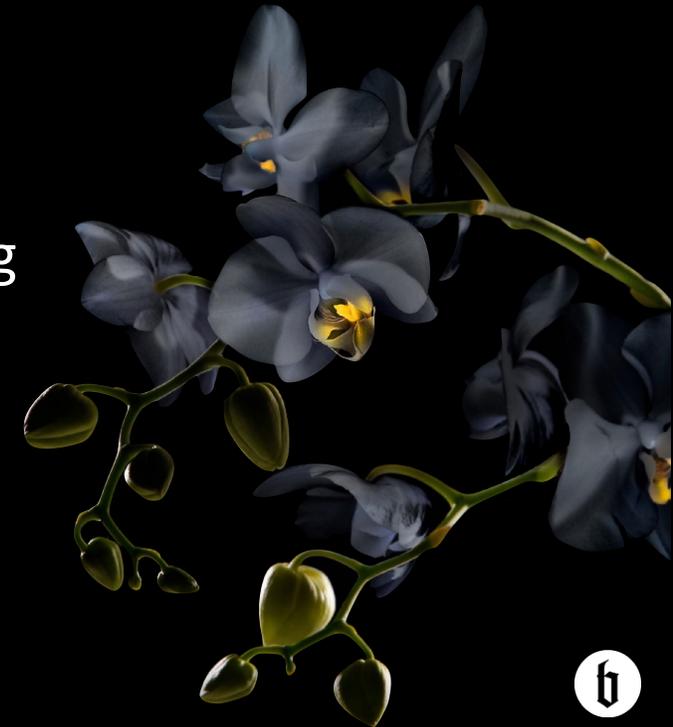
Export anything

- Access any data connected of exported object (incl. relations, dependencies, tags, thumbnails, etc.)



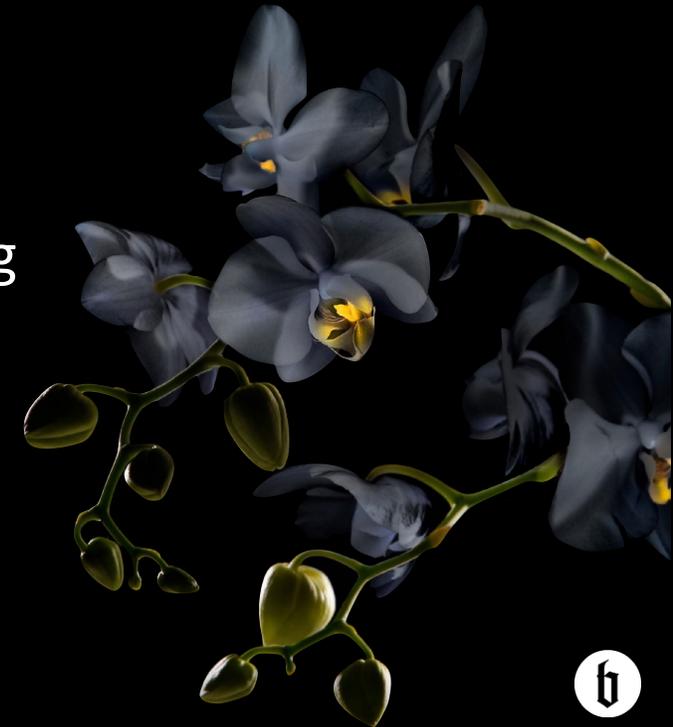
Export anything

- Access any data connected of exported object (incl. relations, dependencies, tags, thumbnails, etc.)
- Export all object brick / classification store fields without listing each single field



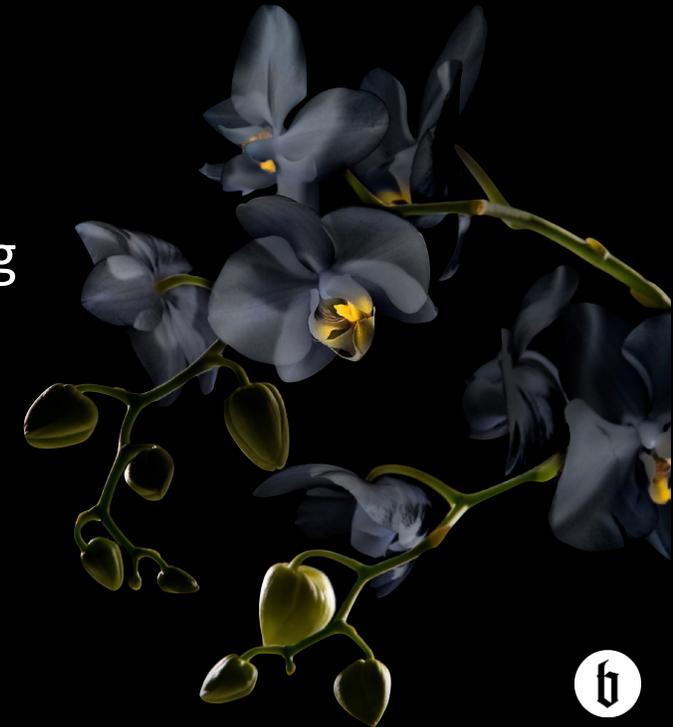
Export anything

- Access any data connected of exported object (incl. relations, dependencies, tags, thumbnails, etc.)
- Export all object brick / classification store fields without listing each single field
- Access data of other objects than the currently exported one



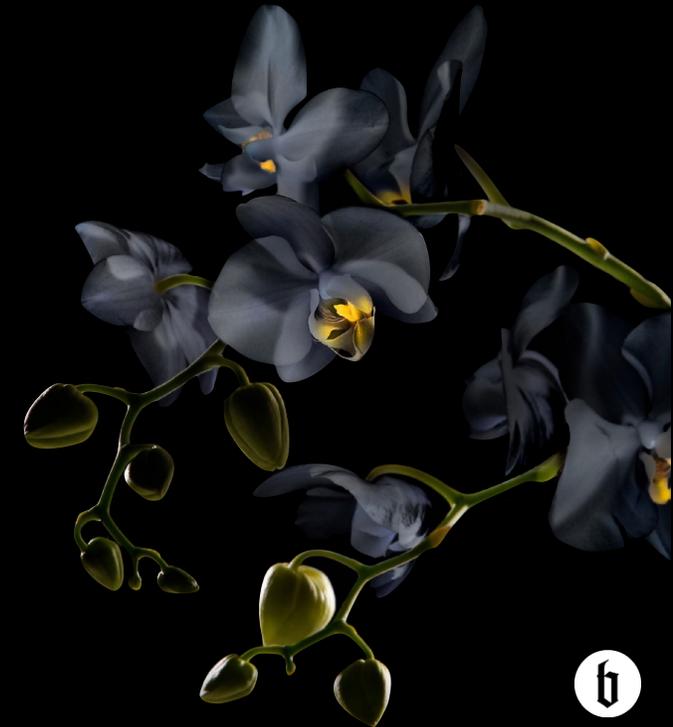
Export anything

- Access any data connected of exported object (incl. relations, dependencies, tags, thumbnails, etc.)
- Export all object brick / classification store fields without listing each single field
- Access data of other objects than the currently exported one
- Preset filters for objects to be exported



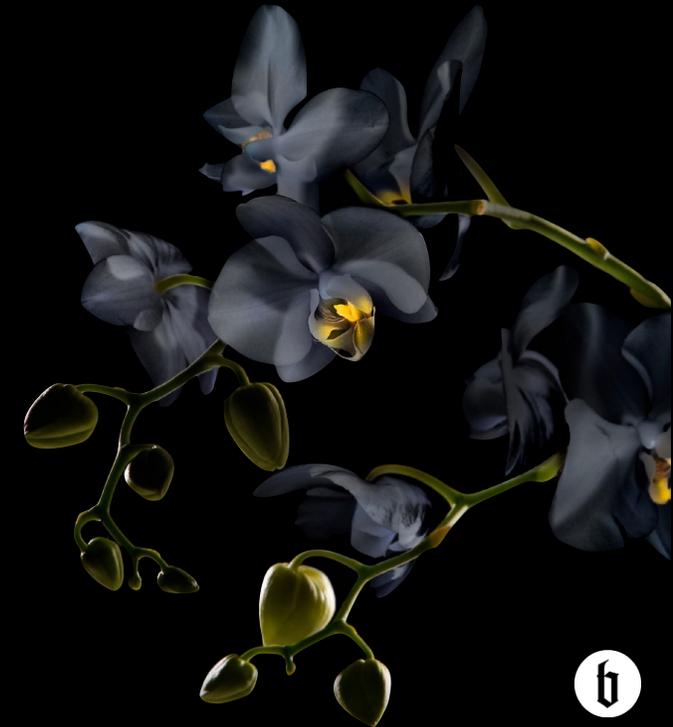
REST API anyone?

- Every dataport can be executed via REST API endpoint



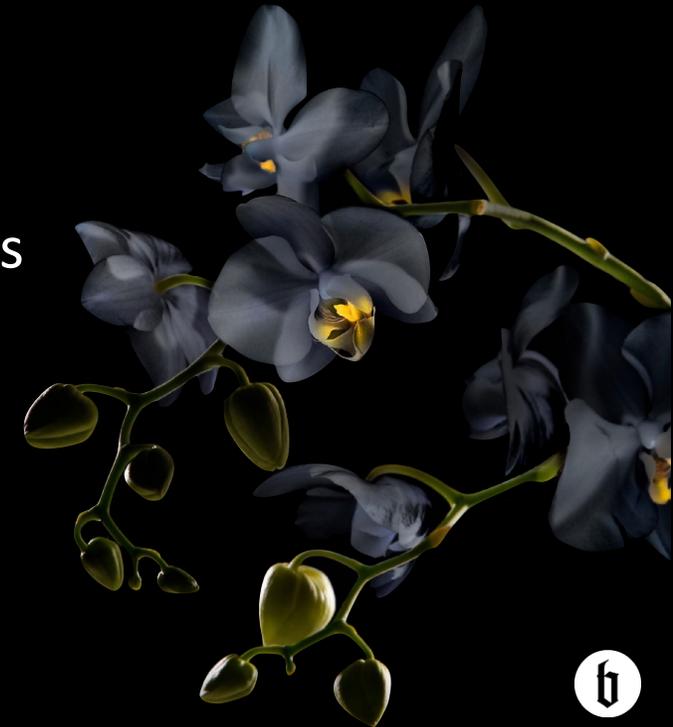
REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Access exports via URL



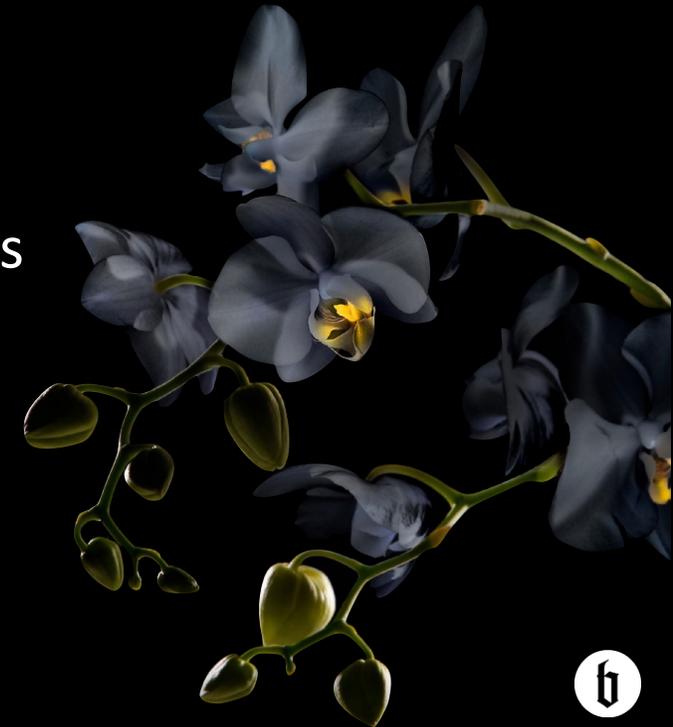
REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Access exports via URL
- → change filter condition / language / logic via URL parameters



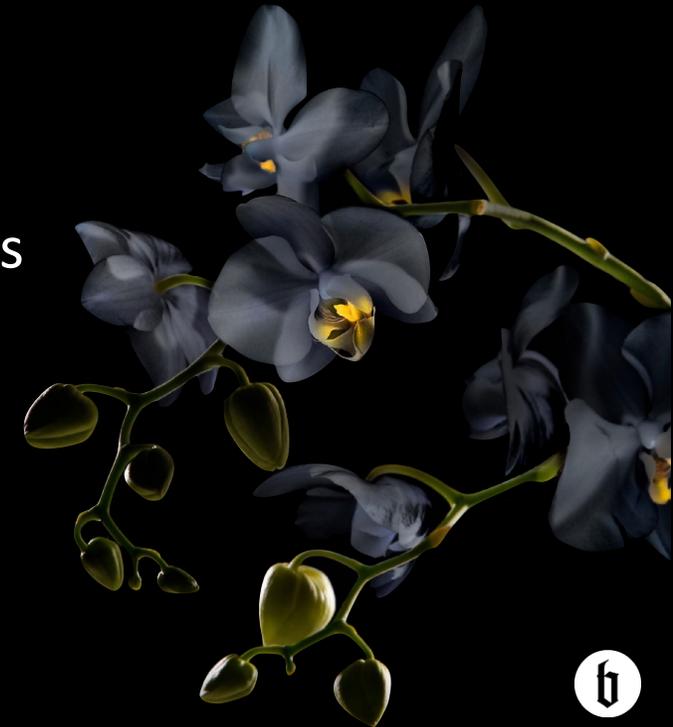
REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Access exports via URL
- → change filter condition / language / logic via URL parameters
- Enter export URL as data feed on platforms like ProductsUp, Google Shopping, FactFinder, Easycatalog (InDesign) etc.



REST API anyone?

- Every dataport can be executed via REST API endpoint
- → Access exports via URL
- → change filter condition / language / logic via URL parameters
- Enter export URL as data feed on platforms like ProductsUp, Google Shopping, FactFinder, Easycatalog (InDesign) etc.
- or use it as real API in your web / infrastructure applications



REST API anyone?

REST API Documentation (Dataport "Export Products")

GET /api/rest/export/Export+Products Execute export "Export Products"

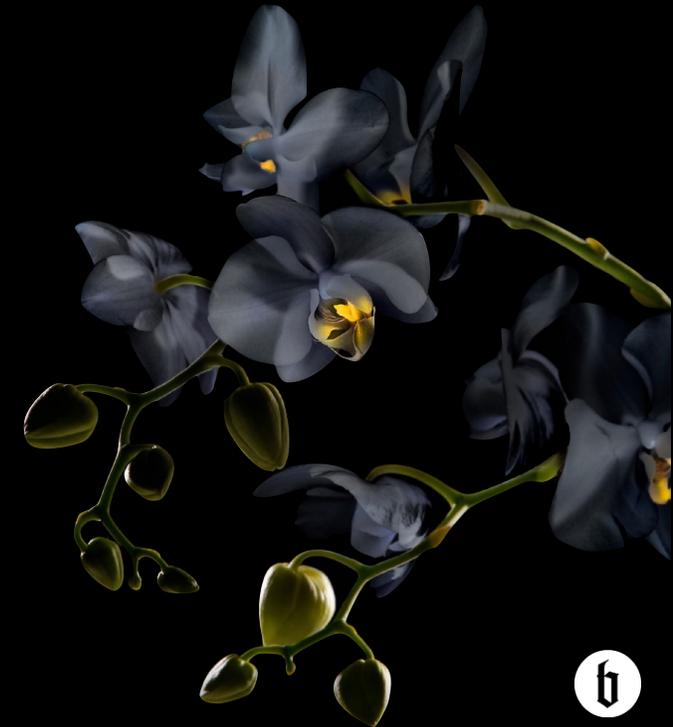
Parameters Try it out

Name	Description
apikey * required string (query)	Rest API key of Pimcore user, see Pimcore Webservice documentation <i>Default value</i> : c9757b1605bb4da4a83b4a3841b94432
	<input type="text" value="c9757b1605bb4da4a83b4a3841b94432"/>
locale string (query)	Locale to be used for localized fields, if omitted the API key's user language gets used. Valid values: en, de <i>Default value</i> : en
	<input type="text" value="en"/>
query string (query)	SQL condition to define which objects to use (this gets added to the configured dataport SQL condition) - add database index to used fields in class definition to enhance performance
	<input type="text" value="query - SQL condition to define which objects"/>
limit int (query)	Max. Number of datasets to be returned
	<input type="text" value="limit - Max. Number of datasets to be returned"/>
offset int (query)	Index of first dataset to be returned
	<input type="text" value="offset - Index of first dataset to be returned"/>
sku (query)	Variable in import resource
	<input type="text"/>



Internal Checks & Transformations

- Imports with same source and target



Internal Checks & Transformations

- Imports with same source and target
- Data Quality Checks



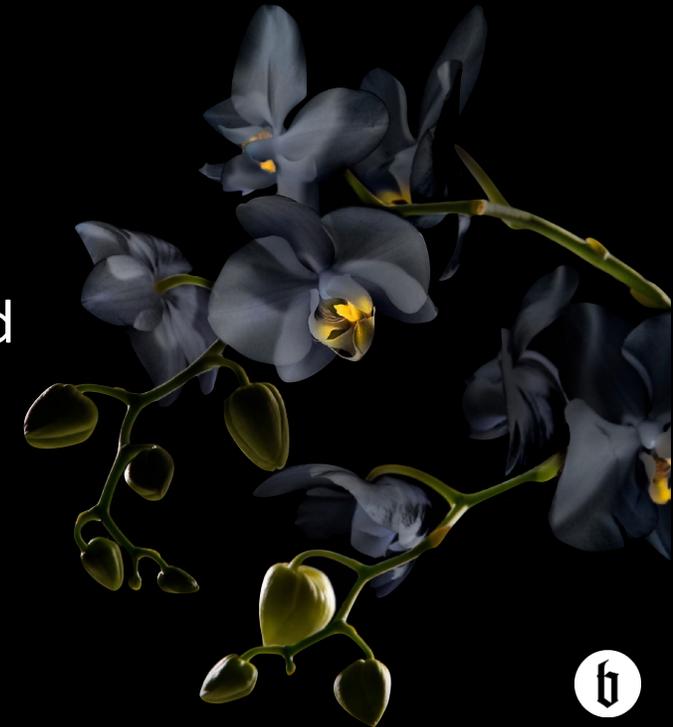
Internal Checks & Transformations

- Imports with same source and target
- Data Quality Checks
 - if product has no price it must not be published



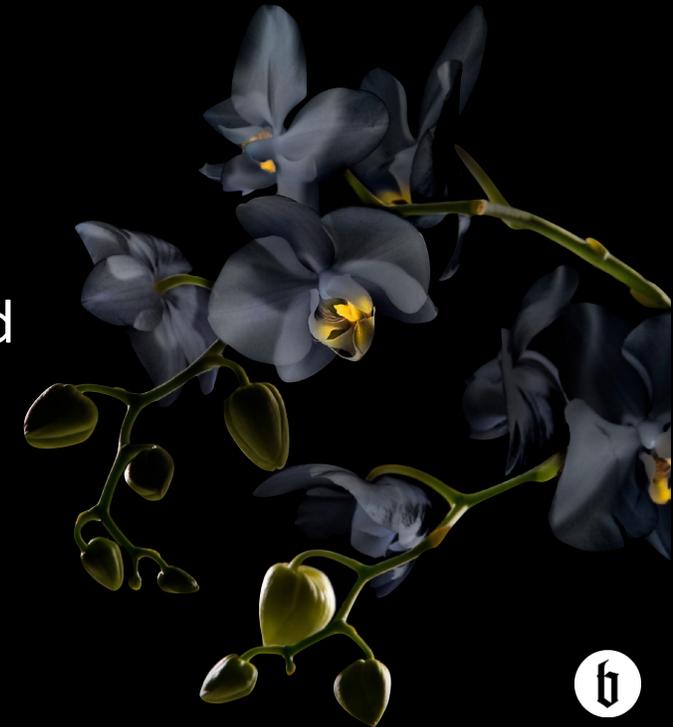
Internal Checks & Transformations

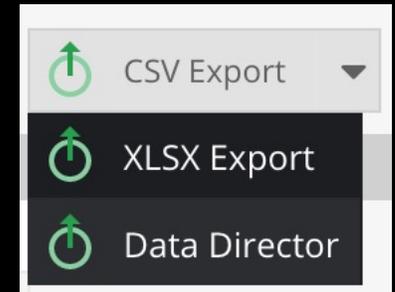
- Imports with same source and target
- Data Quality Checks
 - if product has no price it must not be published
 - if field "price" is not filled and object has not been changed for 1 week → send mail to product team



Internal Checks & Transformations

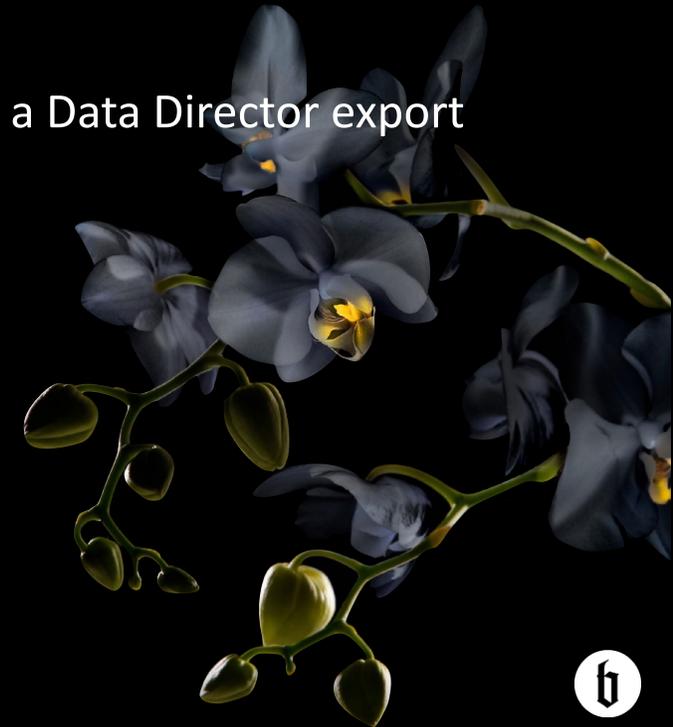
- Imports with same source and target
- Data Quality Checks
 - if product has no price it must not be published
 - if field "price" is not filled and object has not been changed for 1 week → send mail to product team
- Change field types without losing data

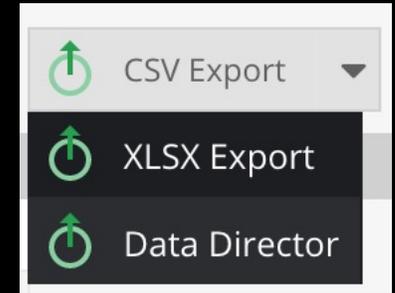




And the really cool things?

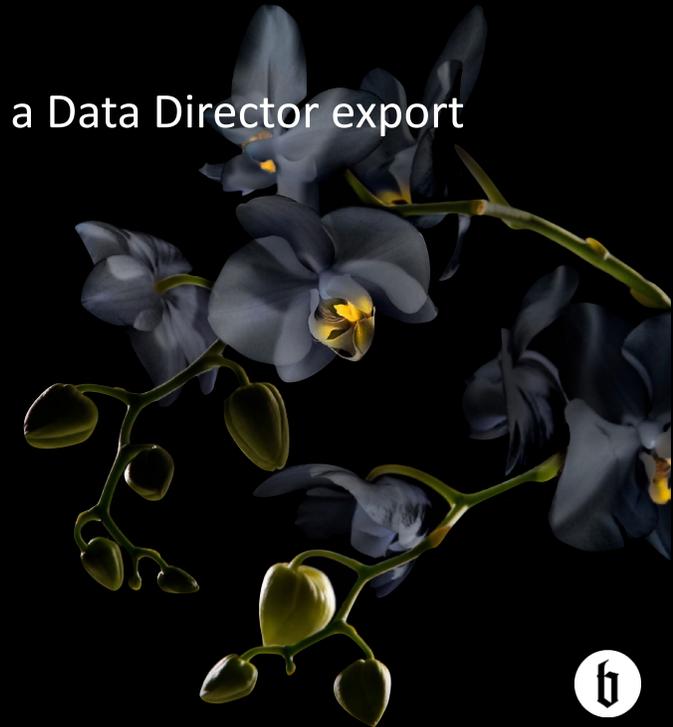
- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export

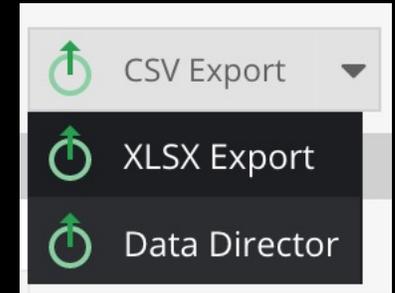




And the really cool things?

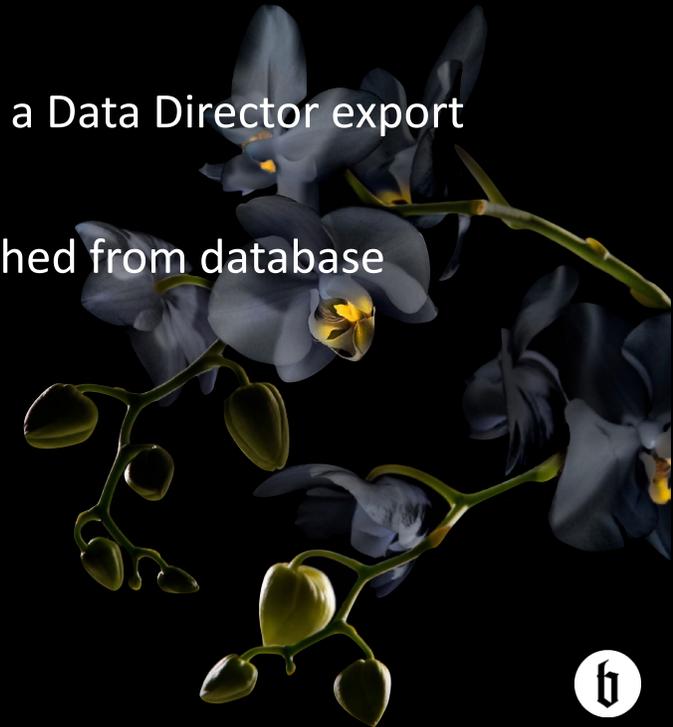
- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export
- Problems with Pimcore's built-in grid exports:

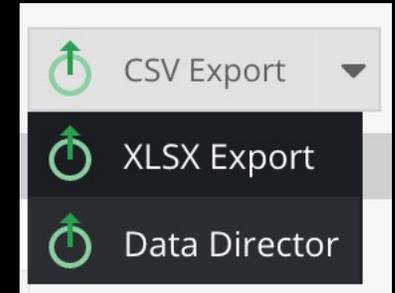




And the really cool things?

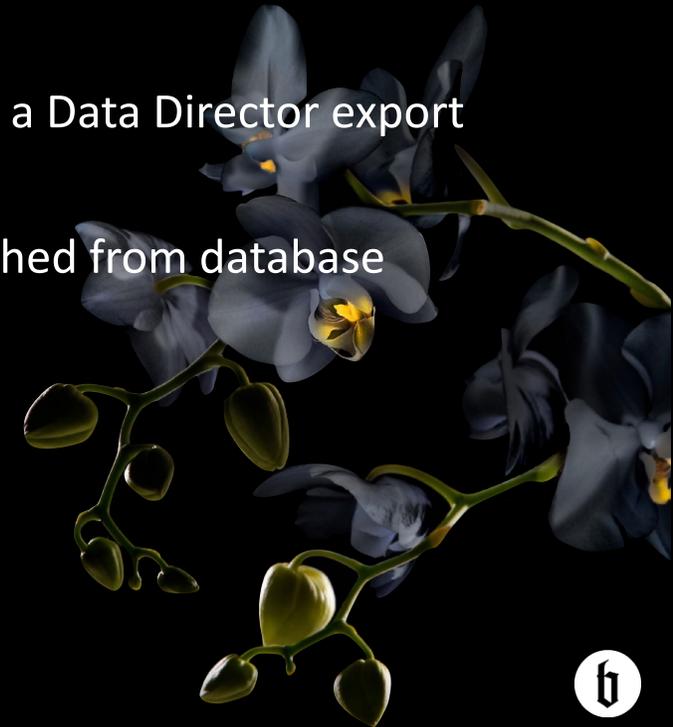
- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export
- Problems with Pimcore's built-in grid exports:
 1. No reuse of previously generated export data -> export data is newly fetched from database when the export is requested -> slow

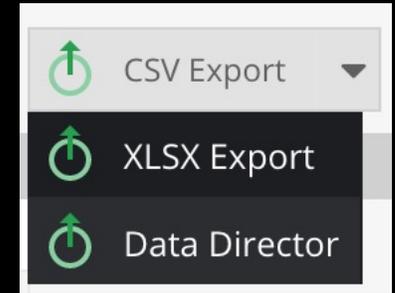




And the really cool things?

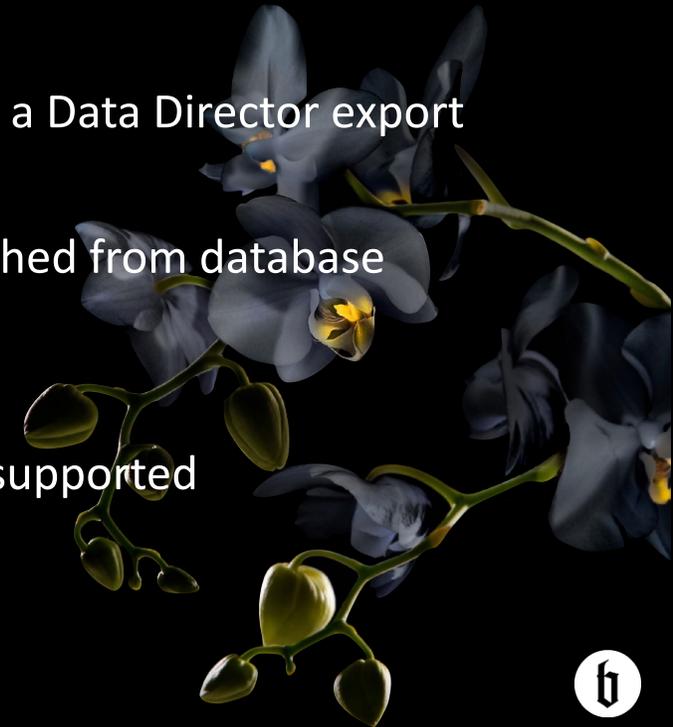
- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export
- Problems with Pimcore's built-in grid exports:
 1. No reuse of previously generated export data -> export data is newly fetched from database when the export is requested -> slow
 2. No predefined filters possible, e.g. to only export published objects

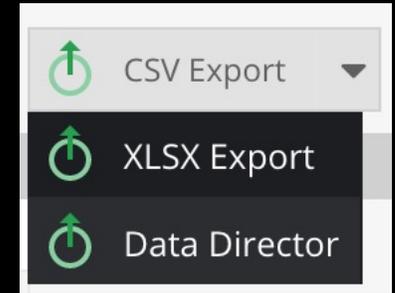




And the really cool things?

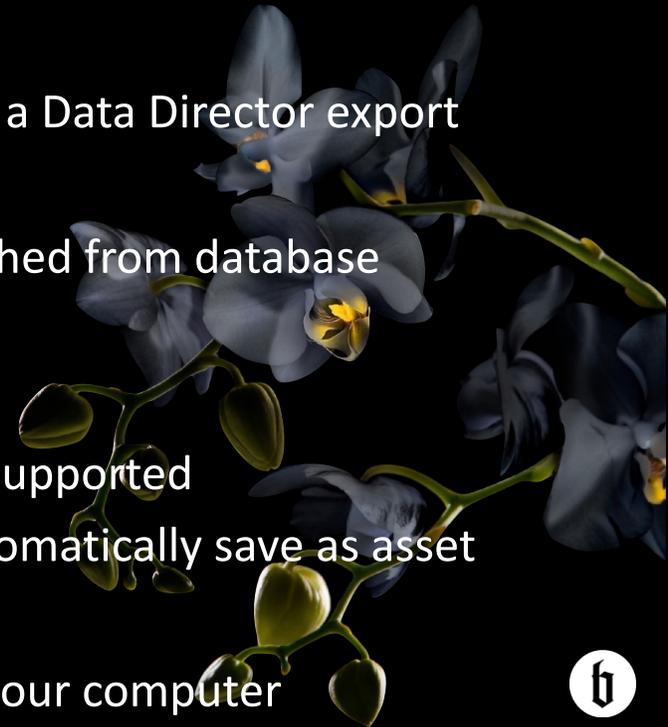
- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export
- Problems with Pimcore's built-in grid exports:
 1. No reuse of previously generated export data -> export data is newly fetched from database when the export is requested -> slow
 2. No predefined filters possible, e.g. to only export published objects
 3. Limited configuration of export format: currently only CSV and Excel are supported





And the really cool things?

- Grid exports
 - use grid view to filter objects → export the filtered / selected objects with a Data Director export
- Problems with Pimcore's built-in grid exports:
 1. No reuse of previously generated export data -> export data is newly fetched from database when the export is requested -> slow
 2. No predefined filters possible, e.g. to only export published objects
 3. Limited configuration of export format: currently only CSV and Excel are supported
 4. Only possibility to download generated export file -> no possibility to automatically save as asset / upload to another server etc.
 - Not possible to start an export and log out from Pimcore / shutdown your computer



And the really cool things?

Data Director ✕

 The structure of the export document and the fields to be exported are only defined in the dataport configuration. It does not matter which fields are currently displayed in the grid. The selected dataport will get executed for all selected / filtered items in the current grid.

Settings

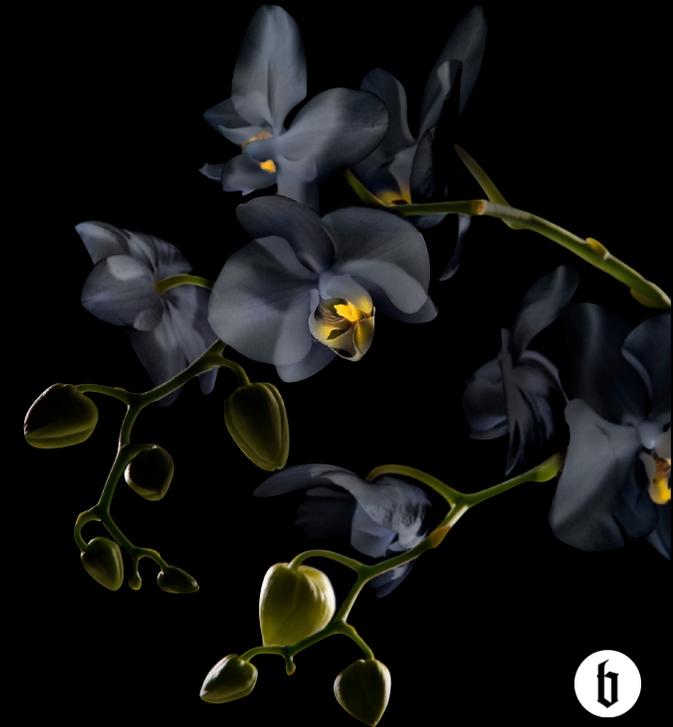
Dataport:

Language:



And the really cool things?

- Every mentioned feature is possible without programming



And the really cool things?

- Every mentioned feature is possible without programming
- But you can always add own logic via callback functions

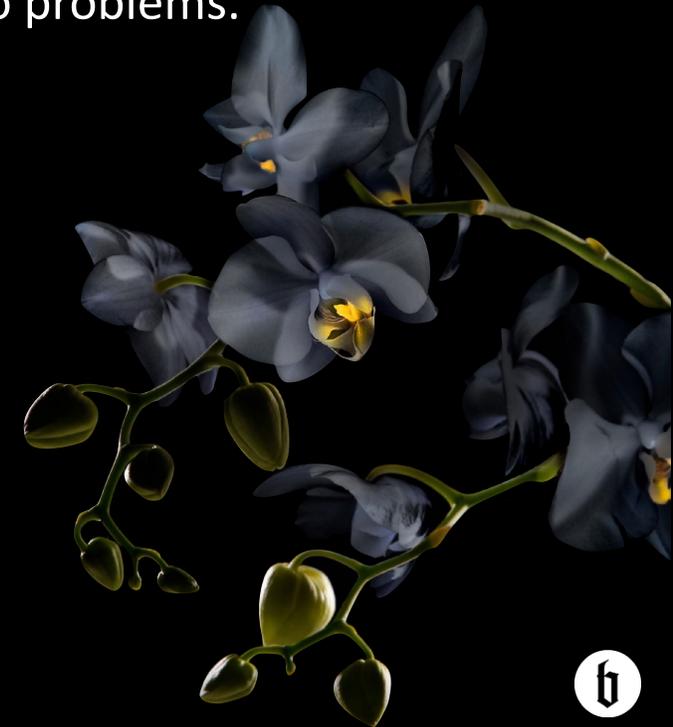


Customer voices

„Our product designers add new products in Excel. As soon as they upload this file to Pimcore, the products automatically get imported, then translated and sent to our ERP system and our webshop. I set this up in a few days and I am not even a developer.“

„This sounds too good to be true – an always up-to-date InDesign export feed, I am dreaming of this for years.“

„We import 8 million products and update 300K daily – so far the Data Director had no problems.“



Customer voices

„Our product designers add new products in Excel. As soon as they upload this file to Pimcore, the products automatically get imported, then translated and sent to our ERP system and our webshop. I set this up in a few days and I am not even a developer.“

„This sounds too good to be true – an always up-to-date InDesign export feed, I am dreaming of this for years.“

„We import 8 million products and update 300K daily – so far the Data Director had no problems.“

**100 %
Customer
Satisfaction ***

* according to survey of all
Data Director buyers 06/21



blackbit

digital Commerce

Pimcore Data Director

Interested?

info@blackbit.de

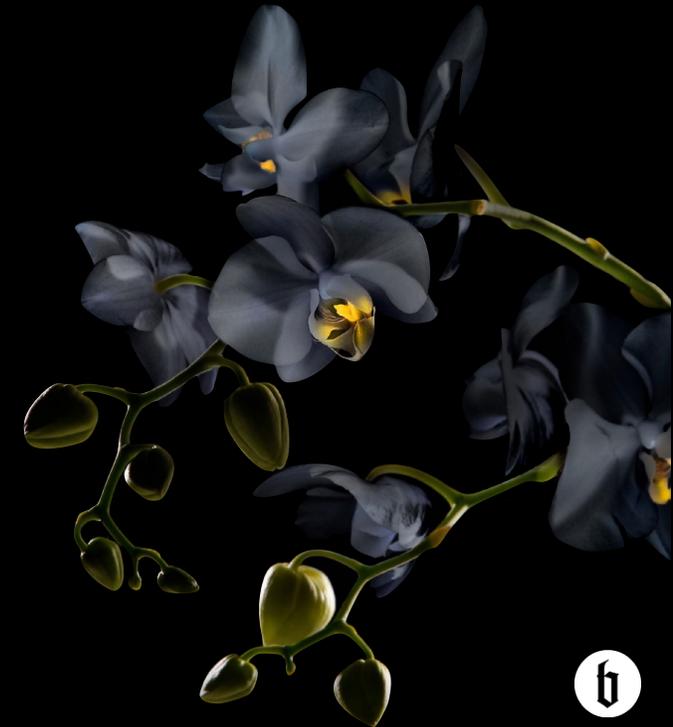
[Manual / Documentation](#)

[Video Tutorials](#)

[Pimcore Demo Access](#)

[Pimcore Marketplace](#)

[Online Shop](#)



PIMCORE INSPIRE' 21

Q&A



Jan
Walther

blackbit

digital Commerce

