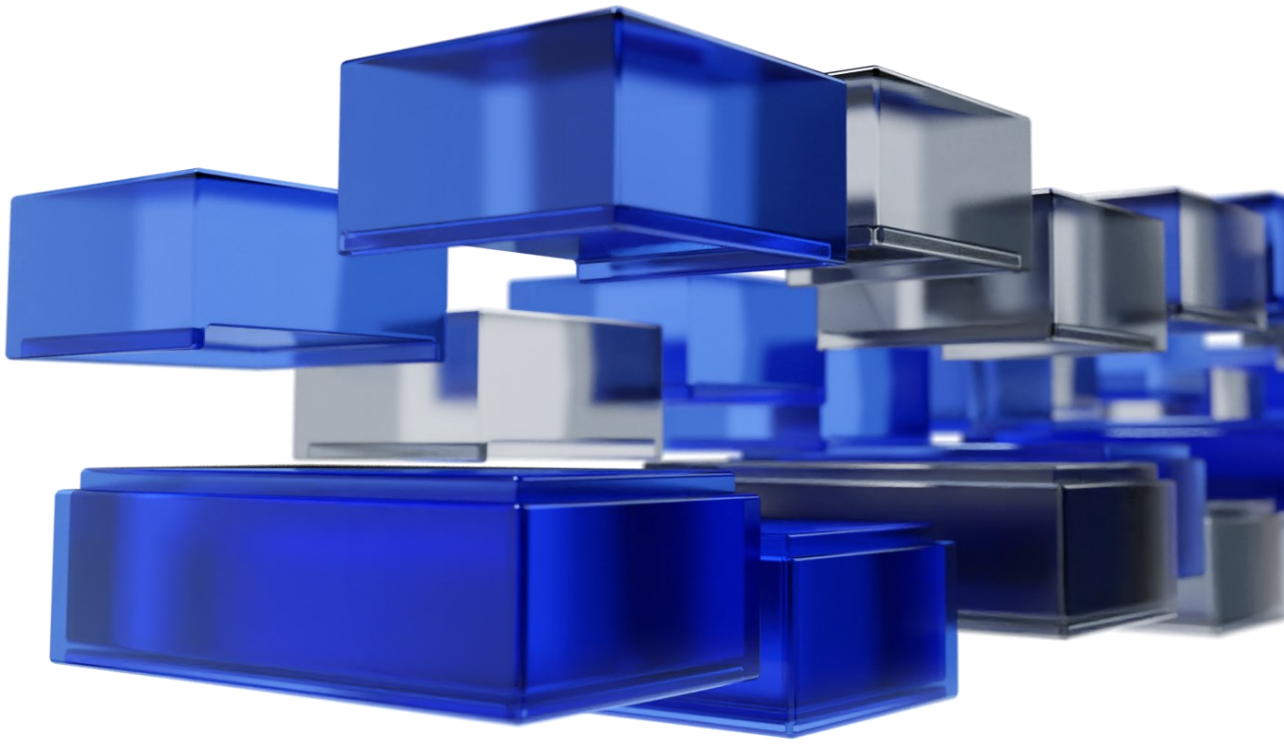
Module building: Best practices



Version:	1.4
Date of version:	October 27st, 2021

Authors can speed up and increase the quality of Module-building by adhering to the best practices the BRYTER team has put together. For more inspiration and to see how these best practices are applied, head to the Templates section or ask your Customer Success Manager for additional tips and tricks.

BRYTER

Table of contents

BRYTER

Frontend/Wizard

1.1. Layout of Input Nodes

Input Nodes are the building blocks of your Modules. Since they will appear on the end-user's display, you should pay special attention to keeping your format consistent and displaying the content in a visually pleasing manner.

1.1.1. General best practices

Headlines

- ☐ Keep formatting style, wording, and syntax of a headline in every Input Node consistent throughout the Module.
- ☐ Headlines should be concise: A headline briefly summarizes the content of an Input Node (e.g., "Personal details"). Refrain from using full sentences and keep it as short as possible with keywords.

First Input Node

- ☐ Use a Module name (e.g., "Commercial Contract Assessor") in your first Input Node in a standalone Module. If the Module is only one component in a Service with several Modules, only use the Module name in the Module that would be used first and do not mention it again in the first Input Node of other Modules.
- ☐ Use *Heading large* for the title of the Module in the 1st Node. Subsequently, use either *Heading large* or *Heading medium*.
- ☐ Do not use a welcome message in the text body of the 1st Node (e.g., "Welcome to the Contract Assessor [...]"). Instead, describe the purpose of the tool and what it can be used for (e.g., "With this tool, you can assess commercial contracts with regard to [...]").
- ☐ Do not use a generic Module name such as “BRYTER tool” but rather explain the functionality, e.g., Assistant, Assessor, Questionnaire, Generator, etc.

1.1.2. Multi-Input vs. Single Input Nodes

Multi-Input Nodes

- ☐ Group a set of inputs that are logically connected, e.g., “Personal Details” or “Address”, into one Node under the same heading.
- ☐ Multi-inputs generally look more appealing as the screen looks less plain and are more user-friendly because they reduce the number of clicks through Single Input Node screens.

Conditional Multi-Input Nodes	☐	When collecting information from end-users that depends on previous input in the Module, use the conditional multi-input feature. This will keep the Module streamlined and ensure the user only see's what they need to see.
	☐	When you need to collect multiple pieces of information from end-users that are conditioned on one or several previous inputs or values, it may be better to use a new input Node for this to reduce the amount of nested or second layer logic.

1.2. Language

Language settings	☐	If possible, select a language & region in the publishing settings (currently English and German) to match the language used in your Module. Note that this setting also affects how numbers are formatted.
Capitalization	☐	Keep capitalization of titles and info blocks consistent.
Tone	☐	Depending on your audience, keep the tone informal or formal throughout the Module and Email or Handover Nodes.

1.3. User experience

Call-to-action	☐	Keep the question or call-to-action close to the actual input field in Single Input Nodes.
	☐	The styling of the call-to-action should be consistent, so either choose headline small or headline medium or keep it formatted in *italics*, **bold,** or ***both***. The format you choose also depends on the theme you have selected.
Info blocks	☐	Decide how you will use info blocks and keep this style throughout the Module as much as possible, e.g., info blocks as hints or info blocks for definitions.
	☐	Depending on your Module's theme, format the headline of an info block as either bold or regular. Apply the formatting consistently throughout the Module.
Images	☐	When using images, use the aspect ratio 2:1 and keep the aspect ratio the same for all images.

	☐	Keep the color tone of images consistent or only use black and white images.
Tables	☐	Use a table when displaying a collection of information to the user. For instance, showing a summary of inputs or outputs using a table will produce a neater front-end that is easy to read and compare. Ensure the titles of the columns in the table are bold to distinguish them from the rest of the information in the table.
Emojis	☐	Depending on the intended audience, emojis can be used in Modules where appropriate. This can make Modules more visually appealing and more intuitive to use.
Videos	☐	Use videos as early as possible in your Module. This will decrease the skip rate.

1.4. User navigation

Content Nodes	☐	Use Content Nodes when no end-user input is required. For example, Content Nodes should be used in the first screen, which provides a summary of the tool and its value proposition.
	☐	Use content Nodes to structure your Modules and provide preliminary results – especially in long Modules.
Confirmation page	☐	A confirmation page in a Result Node should only be enabled if you want to give end-users a visual indication that they have entered all necessary inputs and want to allow them to click on the back button. Note that enabling the confirmation page might encourage users to not complete the Module and drop-off too early.
Linkbacks	☐	Use linkbacks to allow your end-users to go to a certain input Node, e.g., a selection menu. Linkbacks will overwrite the content that was provided when skipping back to a certain Node, so use them only in Modules where this does not affect the Module stats adversely.
Handover Nodes	☐	Always provide clear instructions in both the upper content field and the lower email action. The upper content field should state what the end-user reaching this Handover Node can expect, e.g., who will now be informed, how long will it likely take, etc.

BRYTER

Redirect Results	☐ Use a Redirect Result in combination with a Single-Select Node to allow users to easily restart the Module (Redirect) or conclude the session and Module (Result only). ☐ Use a Redirect Result instead of a link in the content field when redirecting to an external URL, e.g., an intranet page or official guidelines at the end of your Modules.

BRYTER

Backend/Editor

2.1. Initial values

Grouped initialized values

- ☐ For lengthier, more complex Modules where variables may be repeatedly updated depending on various conditions in the logic of the Module, a separate group should be included at the beginning of the Module. This group is used to initialize or introduce the variables that are used throughout the Module, so that these empty variables can later be updated in the Module.
  This is placed at the beginning of the Module, just after the '🚀 START' and before the first input Node.
- ☐ A Node group of initialized values has the advantage of creating a single point of reference for the relevant variables. Users can easily refer back to the initialization group if they wish to amend or update a particular variable name. The effect of this will be to update variables value automatically, rather than having to search through the Module to find the variable or update it in multiple places.
- ☐ Place this group between two Empty Action Nodes to name the group and close it without including any other Nodes. This will speed up maintenance later on.

Author or tester values

- ☐ Use a group of initialized values to introduce values that are only used for testing purposes, e.g., email addresses or names of the testers. Like this, you can simply change OR delete these values in one go and don't receive emails anymore. Quick Check or the Module overview will highlight all deleted values as "unknown variable" which can be easily and quickly deleted when you are ready to publish the Module to LIVE.

2.2. Graph structure

Empty Action

- ☐ The structure of your graph should be as clear as possible. Using Empty Nodes helps to break down complex transitions and should be used whenever you have more than four transitions leaving a Node. This will also allow authors to add Nodes or reconnect Nodes easier.

	☐	Empty Action Nodes help with structuring a BRYTER Module into sections that can be named. Empty Action Nodes also function as the titles for Node groups.
Node groups	☐	A BRYTER Module in its "collapsed state" should ideally only consist of collapsed groups that contain many Nodes. Think of Node groups as the primary structuring element which is always visible - the Nodes contained in a Group are only visible on demand.
	☐	There should be only one transition leading to/from a group.
	☐	A Node group should always include an Empty Action Nodes as its first and last Node - the naming of such empty entry and exit Nodes should be consistent and easy to read, e.g., ALL CAPS (e.g., "INITIALIZATION").
Results Nodes	☐	Result Nodes tend to extend the graph horizontally and makes scrolling cumbersome. Keep the number of result Nodes to a minimum and use updated Text Block Value Nodes to show different content.
	☐	Group outcomes of your Module as much as possible, e.g., Next Steps, Redirect, Report, Guidance, etc. Consider using Content Nodes instead to display the content or file downloads and funnel them back into a few (Redirect) Result Nodes.

2.3. Node titles

Naming	☐	Node titles or names are labels in the top right of the WYSIWYG panel and what is visible in the graph as the name of a given Node. The naming of variables in BRYTER should always be **idiomatic** in the sense that anyone viewing the graph can easily get an idea of the content and function of the variable at first glance.
	☐	It is also crucial to adopt a **structured approach** (per Node type or across the entire Module) to naming them so that, especially in larger Modules, the overview doesn't get lost.

	☐	There are three common ways to name variables (with the first two being the recommended BRYTER): 1) Underscore: All lowercase and whitespaces replaced by underscores - e.g., **personal_details**. This is recommended over camelcase since it is a good balance between readability and technical necessity. 2) Camelcase: First word lowercase and then without whitespace but beginning with capital letter from 2nd word onwards - e.g., **personalDetails**. 3) Spaces: Each word beginning with a capital letter or each word lowercase and with whitespace between words. Spaces are generally not recommended in more complex Modules with longer variable names or multiple similar variable names where a differentiating naming convention may be required, as underscoring or Camelcase make small, systematic differences between names more readable/clearer.
	☐	Instead of phrasing variable names as questions (e.g., **Location of company?**), a prefix can be used to mark a variable as a question (e.g., **ask_location_company**). In the given example, all questions then would have the prefix **ask_** . Similarly, dates could have the prefix **date_** (e.g., **date_deadline_end**), numeric values, or currencies - depending on context - sth. like **amount_** or **eur_** .
	☐	If there are several single pieces of information for the same main object, e.g., the name, email, and address of the employer, a suffix shall be used to categorize such information and make clear it belongs together at the same time: **employer_name** , **employer_email** , **employer_address**.
Placeholders	☐	Ensure that placeholders in Templates correspond to Node titles to enable auto-mapping and quick prototyping.
Empty Action names	☐	Empty Actions can have different purposes and hence different naming styles applied to them. The names, in this case, should nevertheless still be short and clear to summarize the purpose of the Empty Action: 1) If an Empty Action is only used for structuring the graph, e.g., by unifying several paths into one, it can simply have a - as their variable name since they do not need to be distinguished amongst others.

2) If an Empty Action is used for opening/closing a group (→ see above), it shall be UPPERCASE - whitespaces are more appropriate between the words in this case, and the names of the Empty Action should be short and clear to summarize the function of the group.

Database titles	☐	Use a clear title for Database Action Nodes to indicate if this database is reading out of a database or writing/updating content, e.g.,, “Write into Report DB”.
Services components	☐	Always title all Service components and use clear names for all Modules, Case Databases, and Data Views. This will be especially important whenever you need to use write/read Database Action Nodes or set a link to other Module or Data Views.

2.4. DRY (Don’t repeat yourself)

Info blocks	☐	Especially in large Modules, you might want to provide specific information (simple value or whole elements such as **Info Blocks**) repeatedly throughout the Module. In such a case, you must not repeat such information in every place it is used *by its value* but rather have the value in one central place (at the beginning of the Module) and then reference it further down (usage *by reference*).
URL Parameter Configuration Group	☐	Furthermore, some information or values might be injected via URL parameters from outside into the Module and hence there will be **URL Parameter Actions** at the beginning of the Module. In order to group such configuration/information settings, a Config Group (or Info Group) should be placed at the very beginning of the Module – even higher up than the INITIALIZATION group.
	☐	In this config group, settings that are derived from the value of incoming URL parameters are set at this point whenever possible. The globally configured values (**Text Blocks, Calculations,** etc.) can then be **@referenced** further down. If the value itself needs to be changed or another case added, such change only needs to be made in the Config Group - the @references will automatically infer the updated or added values.
User details	☐	For Modules that are customized, e.g., the end-user is addressed by name and receives customized assets, the Input Node asking for end-user details early (high) in the graph, so the values can be referenced at any stage in the Module.

2.5. Complex conditions

Usage by reference

- ☐ Wherever possible, complex logic within the graph shall only be defined and used once in a single place. Further down, such complex logic shall not be repeated by value, but only referenced to *(usage by reference).* That way, errors stemming from repeated Conditions, which are broken, forgotten or contradictory, can be prevented. To achieve such reference to a condition, it shall be broken down to a binary logic (*fake Boolean*) using a Calculation (e.g., named **condition_is_met**). This is first set to **0** (assuming the Condition is not met) and only if the Condition is fulfilled, the value is set to **1**. Further down, the full Condition is then not repeated again when it is needed, but it is only checked, whether **condition_is_met** has the value **0** or **1**.

Naming conditions

- ☐ Generally, authors should avoid naming transitions for simple conditions and rely on the automatic naming to minimize errors.
- ☐ For complex transitions, it makes sense to name conditions using the label box in the top right of the sidebar.
  This makes the Graph more readable (since the complex Condition won't be displayed properly there anyway), and allows to debug such conditions easily using the Module Overview feature. Only if the Condition is given an explicit name, it will show up in the left column – otherwise, this column will be blank.

2.6. Documentation and maintenance

Notes

- ☐ Notes are particularly helpful to display information to other Authors or anyone who will need to maintain the Module. Notes can be great to highlight to-dos or revisions that are required or have been added to the Module.
- ☐ Use notes in Calculation Nodes to display the formula used in the Node or if a custom number format is applied.
- ☐ Use notes in Date Value Nodes if a custom format is applied.

Module Summary/ Info

- ☐ The Module Summary is a great way to explain the Module and indicate the Module owner/originator and important dates for publishing or revisions.

	☐	Toggle the box “Show when opening this Module” to ensure the Module Summary pops up every time an author opens the Module if it contains important information or a Read Me section.
History	☐	Leave short descriptions (max. 50 characters) to indicate significant changes to your Module or which author published the Module to Test/Live.
Module Overview	☐	Use the Module overview to see Notes at one glance or to quickly update certain Nodes or Transitions.
	☐	Untoggle “Compact view” to see all content, fields, and notes at once.
Images	☐	Don’t drag-and-drop images into the content field. Instead, use the insert drop-down and consider linking to the image URL rather than uploading from your local drive to improve loading times of Modules.

BRYTER

Document Automation

3.1. OVO vs BYOT (One-value-only versus Bring-your-own-template)

One value only

- ☐ Whenever formatting such as font, size, or color or the omission of empty lines, rows, and columns is not necessary, authors can choose to simply reference all content into a Text Block Value Node called RESULT.
  This Node then needs to be mapped to the only placeholder in a template. This template could either be the built-in BRYTER template or a custom template with only one placeholder called {{RESULT}}.
- ☐ This approach is usually best suitable for reports sent to the author or summaries that are not made available for clients.

Own template

- ☐ If the intended audience of a generated document is a client or anyone external, you should use your own Word template and use more than one Text Block Value Node to be mapped onto placeholders.
- ☐ If you want to have more control over formatting text such as using different fonts, text sizes or colors and avoiding empty lines, you should use your own Word template with placeholders formatted accordingly or placeholders followed by commands such as {{\dl}}.
- ☐ If the Module's main purpose is generating a document, start with preparing the template for automation. After all dynamic placeholders are defined, build the document automation Module.

3.2. Placeholders

Quick templating

- ☐ If your document template already contains placeholders or indicators for dynamic content, for example in square brackets [], use the Replace functionality to **Find [** and **Replace with {{** - and respectively, **Find]** and **Replace with }}**.

Word Add-in

- ☐ Use the BRYTER Microsoft Word Add-in, to add placeholders and commands quickly.

Naming	☐	The placeholders in document templates for variable values should have the same name as the variables in the Editor. If the Module finds an exact match, it will give you the option to automatically connect the placeholder sections in the Module with the document placeholders.
	☐	For variables associated with document generation and document placeholders, underscoring {{**company_name**}} or Camelcase {{**companyName**}} should be used to avoid missing any placeholders due to accidentally placed whitespaces, e.g., {{**company Name**}} and {{**company Name**}}.
	☐	If spaces are used in names and placeholders, enable the ¶ functionality to spot whitespaces quickly.
Commands	☐	Use command {{\dl}} to avoid empty lines in paragraphs in your Word template.
	☐	Use command {{\dc}} to delete a column or {{\dr}} to delete a row in a table. Note: This only applies to tables inserted directly through the built-in table functionality in Word. Tables created inside the WYSIWYG cannot be dynamically adapted.
Avoid empty lines	☐	If some of the values in your Module that are mapped onto placeholders are skipped due to the conditions defined in your Modules, make sure to add the delete line {{\dl}} command after the placeholder in your document if the placeholder is the only text present in that line.

3.3. Grammar groups

Dynamic inline references	☐	A Grammar Group can be used to ensure that the output document is grammatically sound and customized. This is especially useful when automating forms which specifically address a person whose gender is known. Instead of using hard coded variants (e.g., "the employee shall use his/her own computer"), use a dynamic inline reference using an @reference to an updated Text Block Node called “His/Her/Their”. Depending on the selected gender this Text Block Node is updated with the correct pronoun which leads to fewer Text Block Nodes.

Naming	☐	Use an idiomatic Value name like His/Her/Their instead of something more descriptive such as personal_pronoun_third_person_singular.

BRYTER

Case Databases and Data Views

4.1. Setup

Case Databases initial setup	☐	Consider creating the Module first to ensure all values can be correctly mapped into fields of the case database.
	☐	Clearly indicate if a Case Database Action Node is reading out of a Database or writing values into the Database in the Node's title.
	☐	Use an idiomatic name for your Case Databases. If you are collecting information on clients, title your Case Database "Client Details".
Data Views initial setup	☐	Use an idiomatic name for your Data Views, especially if you create several Data Views intended for different audiences to clearly distinguish between them.

4.2. Configuration

Case Databases configuration	☐	Ensure that all or almost all values are clearly defined and have distinct field names. This will also help in selecting the correct type for values.
	☐	Use emojis in field names and short field names if the Service contains Data Views. Emojis and field names can be added or edited after your initial configuration.
	☐	If you are using Data Views to display the status of records in your Case Database add the Status as one of the first fields in the Case Database.
	☐	Collect end-user emails if the Service includes a Data View that needs to be filtered according to the end-user accessing the Data View.
Data Views configuration	☐	Only select the fields that are required in a Data View and deselect system values like "Last update" or "ID" that are not required.

BRYTER

Publishing

5.1. General Settings

Language & Region	☐	Select the language of the navigation and formatting of numbers according to your end-user audience.
Do not save answers for statistics	☐	Select this option only if your Module contains no case databases or integrations and the ability to resume the published Module is not required.
Enable save and continue later	☐	Enable save and continue later in large Modules after analyzing drop-off rates or after user feedback.
Use a custom theme	☐	Always try to use your own themes to ensure the correct licensed font is used and it is brand-compliant.

5.2. Access Settings

Only accessible after login	☐	Use this setting if your Module is intended for internal audiences and SSO is enabled in your tenant or if you have created and distributed user accounts to your end-users.
Only accessible with a password	☐	Use this setting only if there is a secure way to distribute the password and if “Only accessible after login” is not an option because end-users should remain anonymous or creating user accounts would create too much overhead.
Only accessible via an API key	☐	Use this setting if a Module should only be used once or a pre-defined number of times.

5.3. Published Module Configuration

Show navigation	☐	Disable navigation if you want to discourage end-users from clicking back in the published Module.
Show progress bar	☐	Disable progress bar only if the length of your Module varies extremely between different paths and you included linkbacks.
Show scroll assist	☐	Disable scroll assist only if you are using no or very few Multi-Input Nodes.