


Json guide for LCAbyg version 5.0



Title	Json guide for LCAbyg version 5.0
Subtitle	
Published	December 2020
Last updated	December 2020
Authors	Simone Bruhn, Christian Grau Sørensen
Keywords	json, JavaScript Object Notation, LCAbyg
Publisher	BUILD - <i>Institut for Byggeri, By og Miljø</i> , Aalborg Universitet, A.C. Meyers Vænge 15, 2450 København SV Email: sbi@sbi.aau.dk www.build.aau.dk

Before reading the json guide for LCAbyg we recommend you to read the user guide for LCAbyg version 5.0. The guide is in Danish and can be found here:
<https://www.lcabyg.dk/download-program>

Table of content

1	Introduction.....	4
1.1	The Json File Format	4
1.1.1	Working In The Json File Format	5
2	Guide Build Up.....	6
3	Understanding The Basics	7
3.1	Lists	7
3.2	Dictionary	7
3.3	Nodes And Edges	8
3.4	Id's.....	8
3.5	Nodes And Edges Build Up In Lcabyg	9
3.6	The Files And Folder Structure	10
3.6.1	Gen_Dk And Case1	10
3.7	Import_Example.....	10
4	Applying The Basics – A Step-By-Step Guide	11
4.1	Creating Json Files.....	11
Step 1 -	Download Lcabyg 5.0.....	11
Step 2 -	Understanding The User Interfaces.....	12
Step 3 -	Create A New Project Template	12
Step 4 -	Create New Json Files	17
4.1.1	Creating A New Element (Bygningsdel)	20
4.1.2	Create A New Edge Between Category Element And Element (Categorytoelement)	21
4.1.4	Creating A New Construction (Konstruktion)	22
4.1.5	Create A New Edge Between Element And Construction (Elementtoconstruction)	23
4.1.6	Creating A New Product (Byggevare)	24
4.1.7	Create A New Edge Between Construction And Product (Constructiontoproduct)	25
4.1.8	Create A New Edge Between Product And Stage (Producttostage)	26
4.2	Check The Code	27
Appendix A	28

1 INTRODUCTION

This Json guide for LCAbyg version 5.0 is about the technology used behind the program for quicker import and export in LCAbyg. LCAbyg is a tool used to model a Life Cycle Assessment (LCA) which calculates the buildings environmental profile and resource consumptions.

In LCAbyg version 3.2 it is possible to create a third party integration i.e. connecting LCAbyg with a program developed by others than Sbi. In the new version of LCAbyg this integration has been reestablished. In LCAbyg 3.2 the xml file format is used for this integration where in LCAbyg 5.0 the open file format Json is used.

This guide is mainly for those who want to add their own or/and existing constructions (konstruktioner), building products (byggningsvarer) e.g. from previous LCAbyg programs. Since BUILD cannot control the data quality, the user does not have full access to the library behind LCAbyg named gen_dk. Furthermore, it not possible to create your own building parts (byggningsdele), phases (faser) etc. as this data can ultimately not be validated by BUILD.

Before reading the json guide for LCAbyg we recommend you to read the user guide for LCAbyg version 5.0. The guide is in Danish and is available at <https://www.lcabyg.dk/download-program> (Brugervejledning til LCAbyg version 5.0).

1.1 THE JSON FILE FORMAT

Json stands for JavaScript Object Notation and is an open standard for storing and exchanging data. LCAbyg contains a number of json files in which the user, in some extend, can add to. E.g. when creating a construction (konstruktion) in LCAbyg, a json file is made and placed in the representative folder which is linked to LCAbyg, see Figure 1 and Figure 2.

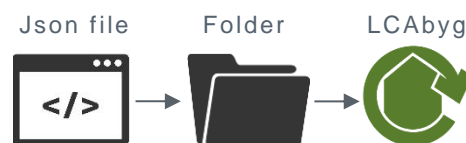


Figure 1 Simplification of how the json files are linked to LCAbyg

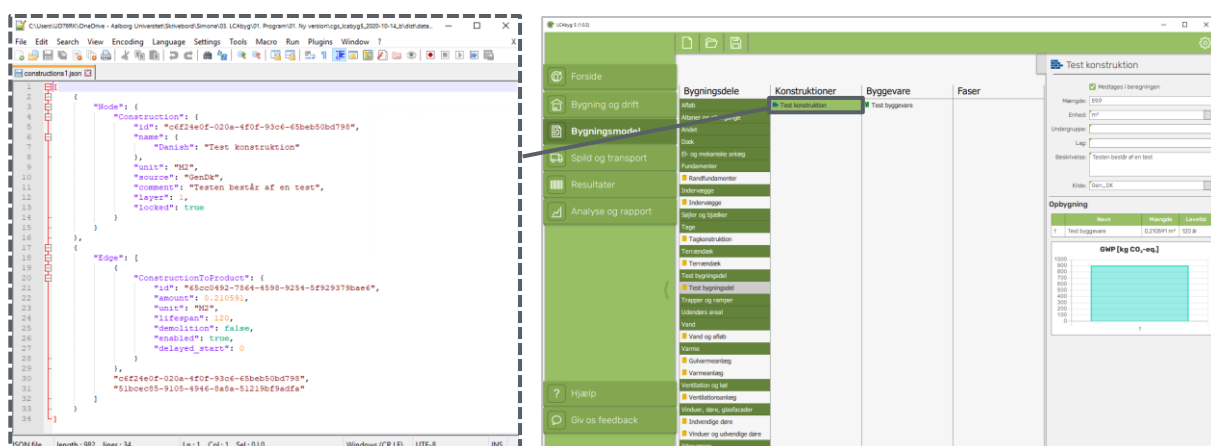


Figure 2 The connection between LCAbyg and a json file. Left: a new construction is made in a json file and connected to a building part. Right: the new construction is visible in LCAbyg

1.1.1 WORKING IN THE JSON FILE FORMAT

A project template named **Import Example** is created with that purpose that the user can work and edit in the json files located in the associated folder (named `import_example`). This guide is based on this project template, and therefore we recommended that you work in this template, to begin with.

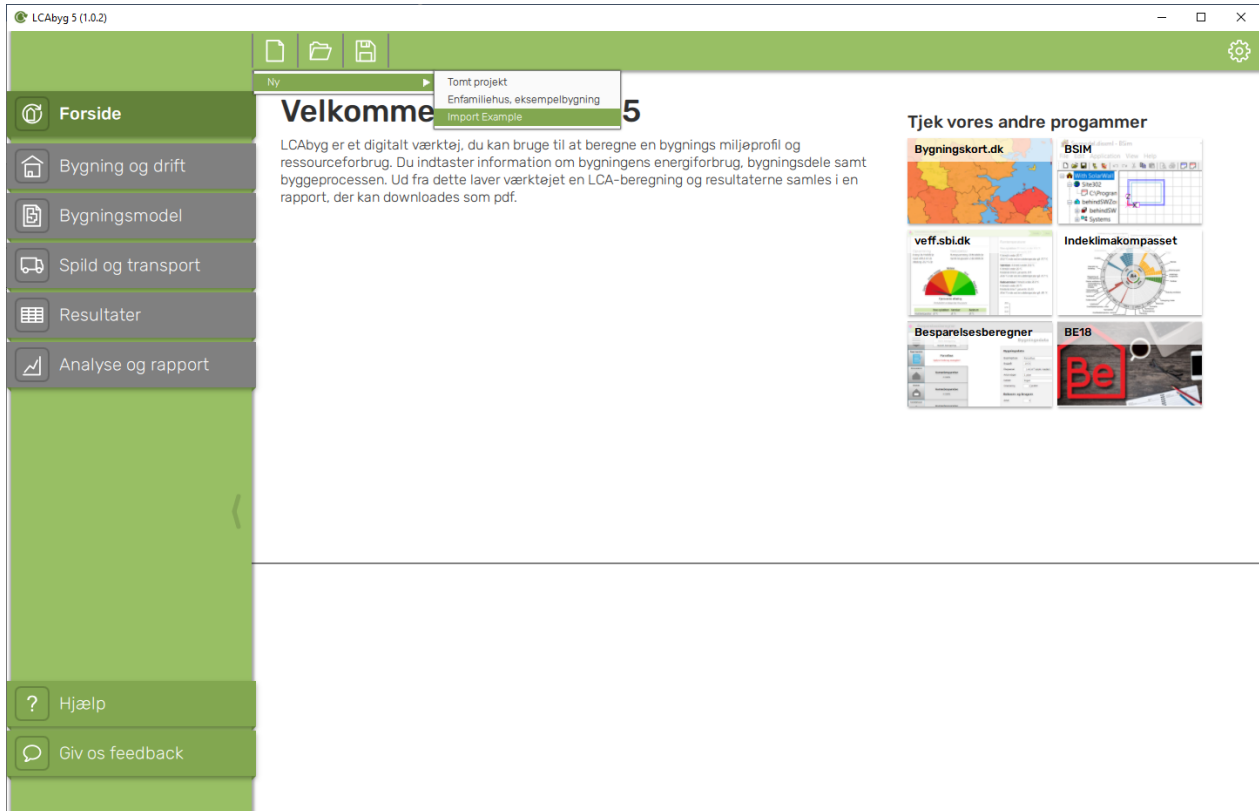


Figure 3 Project template the user can work in in LCAbyg

The project template, `import_example`, is located in LCAbyg 5 folder which can be found in the program folder after downloading LCAbyg. The `import_example` folder holds a number of json files, in which you can work in and add and/or change to, see Figure 4.

Before start working, download a text editor where you can work in the json format. We recommend the text editor Notepad++ which can be found at <https://notepad-plus-plus.org/downloads/v7.9/>

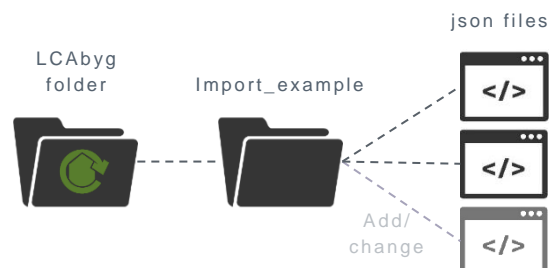


Figure 4 json file location

2 GUIDE BUILD UP

The json guide for LCAbyg is divided into two chapters each containing essential information and knowledge about the json files behind LCAbyg.

Chapter 1, *Understanding the basics*, introduces the basics for the json files for LCAbyg, e.g. how Nodes and Edges works, the folder and files structure, as well as the Nodes and Edges, build up in LCAbyg.

Chapter 2, *Applying the basics*, a step-by-step guide of how to create a new project template in LCAbyg and create new Nodes: constructions (konstruktioner), building products (byggningsvarer) etc. and how to create Edges between the Nodes.

To make it easier for the user to navigate in the json files and LCAbyg at the same time, the Danish term used in LCAbyg, is referred to in parentheses in the headlines.

In the section dealing with the creation of nodes and edges, each page is marked with a Node or an Edge in the upper left corner of the paper to indicate whether the user is creating a Node or an Edge.

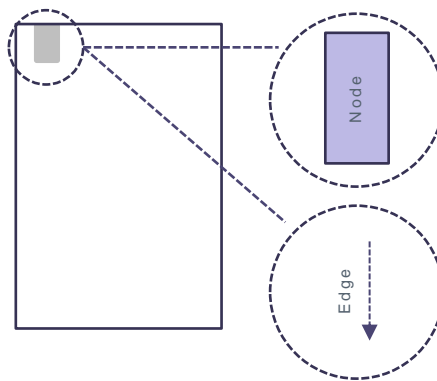


Figure 5 Page indication

3 UNDERSTANDING THE BASICS

The json files consist of lists which contain a number of dictionaries which holds a string of Nodes and Edges, see Figure 6.

When creating a json file, it is important to keep track of the essential characters and indentation, as well as their location in the file. Section 3.1 and 3.2 will review which characters are essential when creating a list and a dictionary.

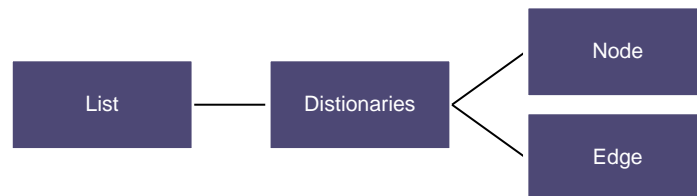


Figure 6 Simplification of the json files built up

3.1 LISTS

The json files consist of lists. Start the code aka the list in the json file by adding “[” in the beginning and end the code by “]” in the end. If you want to add multiple Nodes and Edges in the same code/file this is separated by a comma “,”. The last Node or Edge must never contain a comma. The order of Nodes and Edges does not matter.

Example

```
[  
  Node1,  
  Edge1,  
  Node2  
]
```

3.2 DICTIONARY

The json files contain a number of dictionaries. A dictionary is an unordered collection of data values, used to store data values like a map. The dictionaries are denoted by “{ }” and holds a key/value pair. The keys must be strings and separated from their associated values with a colon.

Example

```
{  
  "Key1": Value1,  
  "Key2": Value2  
}
```

3.3 NODES AND EDGES

The json files consist of Nodes and Edges. A Node can be explained as a visual representation of an entity, where an edge is a visual representation of a relation. An Edge can connect two Nodes, as illustrated in Figure 7.



Figure 7 How an Edge can connect two Nodes

3.4 ID'S

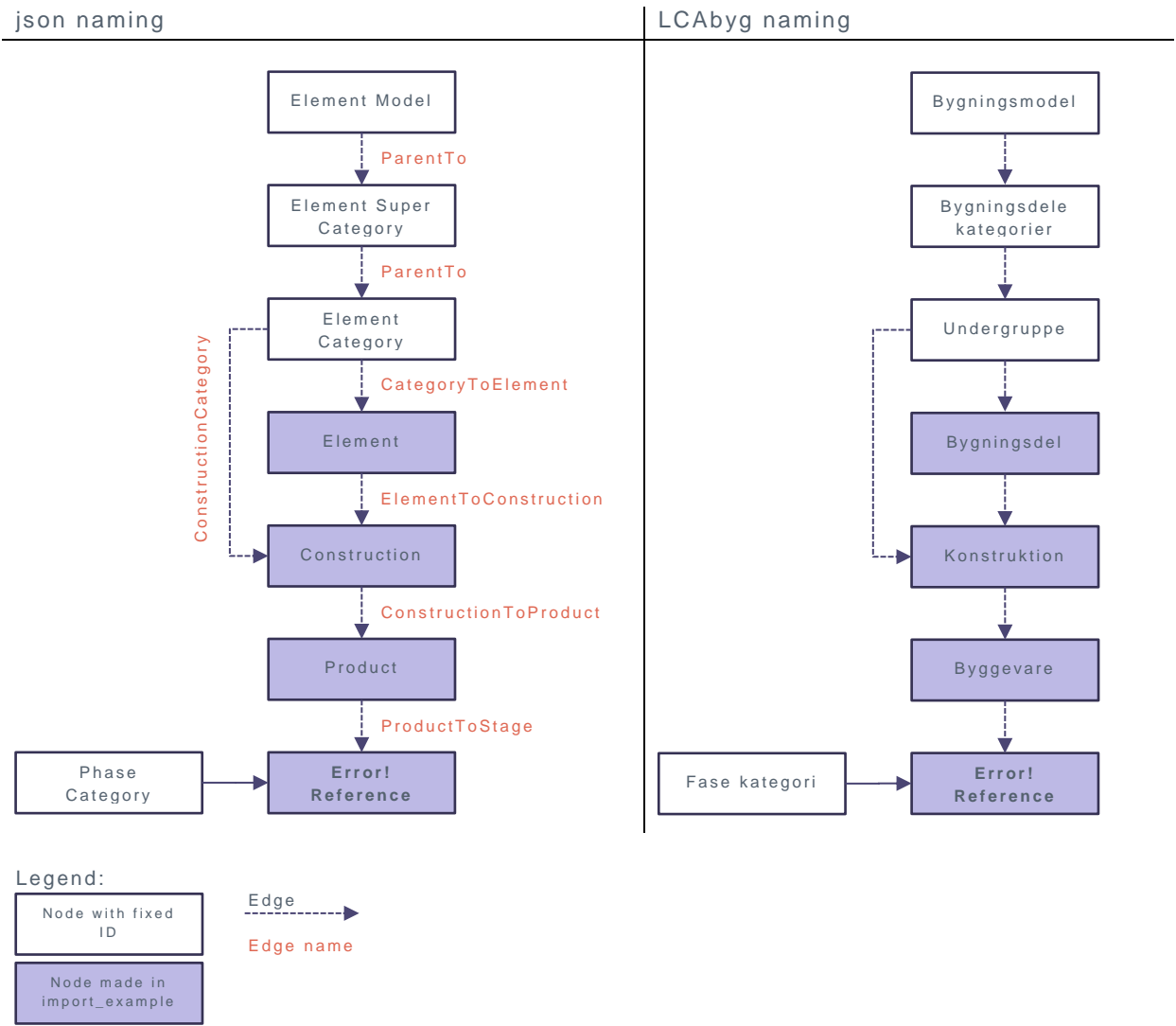
Each Node and Edge contains a unique ID. The ID's follow the UUID standard¹. The ID must only be used once when creating new Nodes and Edges. These IDs is later used to refer to each other e.g. when referring to an Edge between two Nodes (see Refer either existing construction ID or new construction ID). Each time a new Node or Edge are created a unique ID must be created and added. Use the following link to generate unique IDs for the Nodes and Edges: <https://www.uuidgenerator.net/version4>

¹ https://en.wikipedia.org/wiki/Universally_unique_identifier

3.5 NODES AND EDGES BUILD UP IN LCABYG

When creating or adding a new construction, building product, phase etc. a new Node must be created in the json file. The named Nodes in json do not directly correspond to the naming in LCAByg. Table 1 describes what the json Nodes correspond to in LCAByg. As stated before the user can only add new Nodes to some extent. Table 1 illustrates where new Nodes can be added. In section 4.1 a further description of how to create the representative Nodes and Edges is explained.

Table 1 Naming in json vs. LCAByg. All text in the shaded area in the table are linked to a more detailed description and templates to create the specific Node or/and Edge.



3.6 THE FILES AND FOLDER STRUCTURE

LCAbyg consists of a number of folders and json files which define the constructions, products, stages etc., see Table 1.

3.6.1 GEN_DK AND CASE1

The folder `gen_dk`, represents the library included in LCAbyg. The folder contains all constructions, products, stages etc. represented in LCAbyg. The library can be found in two ways, when you create an **Empty project** (Tomt projekt) or choose **case1** (Enfamiliehus eksempelbygning) in LCAbyg, see Figure 8. The difference between the Empty project and case1 is that case1 contains a series of defaults for an example building whereas is "empty" and does not contain these defaults.

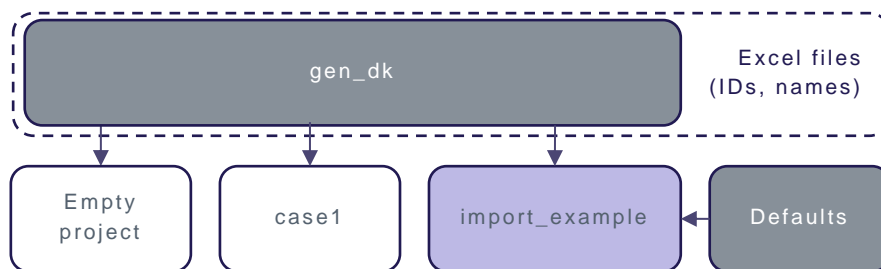


Figure 8 Files and folder structure in LCAbyg

The Excel file

The `gen_dk` folder contains excel files, which all contain the library's IDs and names for the constructions, products and stages in LCAbyg. If you want to make an Edge between a new created Node and an existing Node from the library you must find the library's (`gen_dk`'s) IDs and names in the excel files, see Figure 9 for an example of the setup.

Name	ID
Construction1	111-111-111
Construction2	222-222-222

Figure 9 Excel file built-up

3.7 IMPORT_EXAMPLE

The `import_example` is a project template where the user can add changes or add new building products etc. In the project template, a number of json files have already been added which contain constructions, building materials etc. In the `import_example`, a number of json files (containing examples on constructions, building products etc.) have already been added. When LCAbyg is opened and the project template "Import Example" is selected, constructions, building products, stages etc. can still be imported from LCAbyg's library `gen_dk`, see Figure 8.

4 APPLYING THE BASICS – A STEP-BY-STEP GUIDE

4.1 CREATING JSON FILES

There are multiple steps in the creation of json files. In the following sections, a step-by-step will be reviewed.

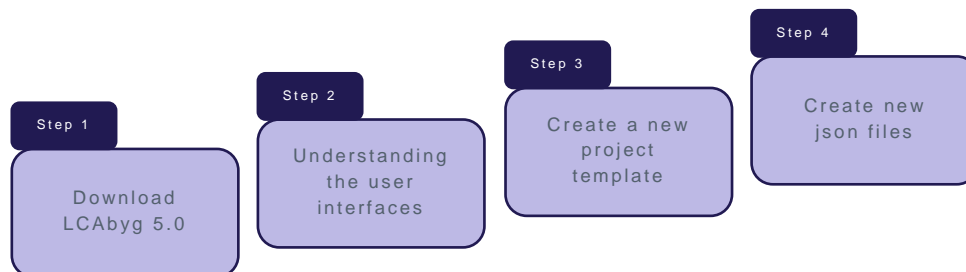


Figure 10 Step-by-step guide for creating json files

STEP 1 - DOWNLOAD LCABYG 5.0

Download the newest version of LCAByg at <https://www.lcabyg.dk/download-program>. You have to create a user before you can download the program. LCAByg 5.0 is only available for Windows at the moment.

If you have an older version of LCAByg, please make sure to open the newest version of LCAByg before working in the program.

STEP 2 - UNDERSTANDING THE USER INTERFACES

Before starting creating constructions, building products etc. in json files, a basic understanding of the user interface in LCAbyg must be established. This guide only touches on a fraction of the understanding of the user interface and a deeper understanding must be achieved when reading the LCAbyg guide version 5. Figure 11 illustrates the basics of the user interfaces in LCAbyg 5.0

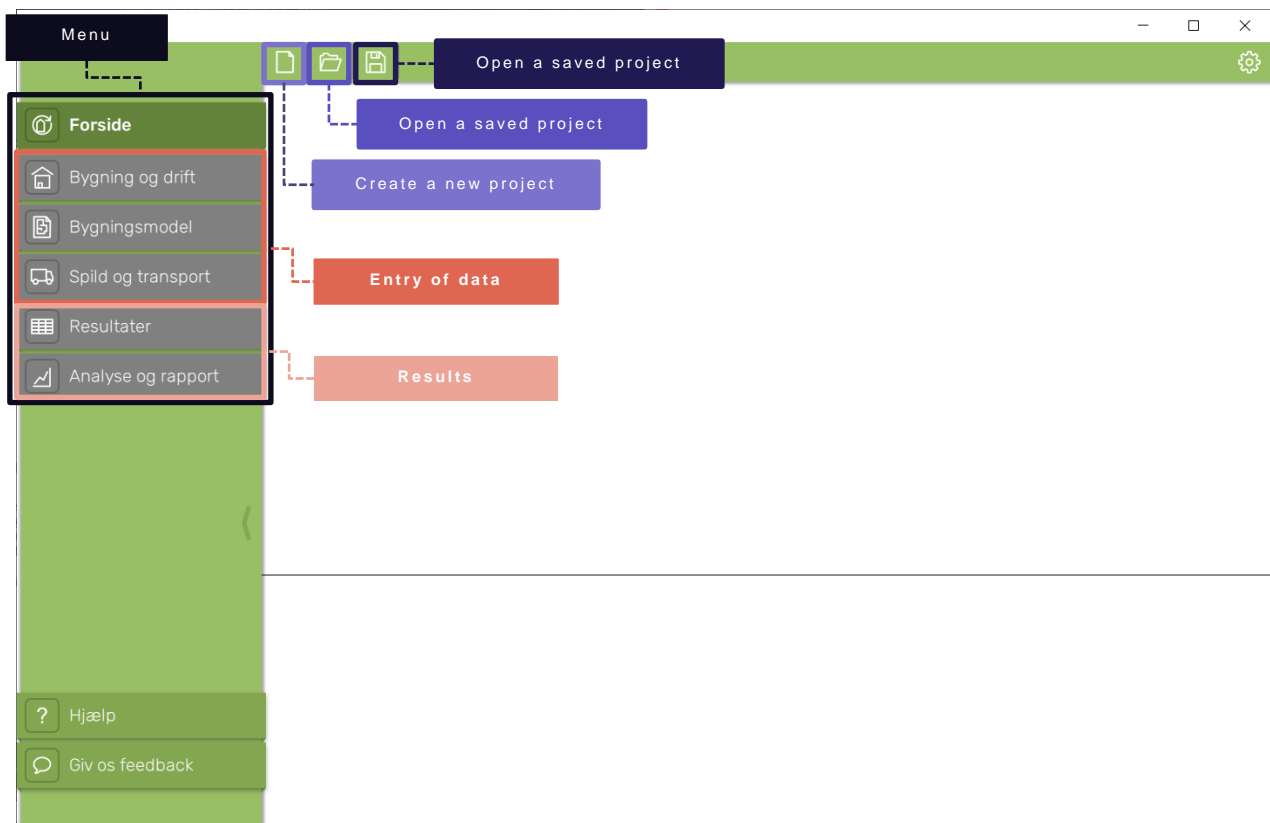


Figure 11 User interface in LCAbyg 5.0

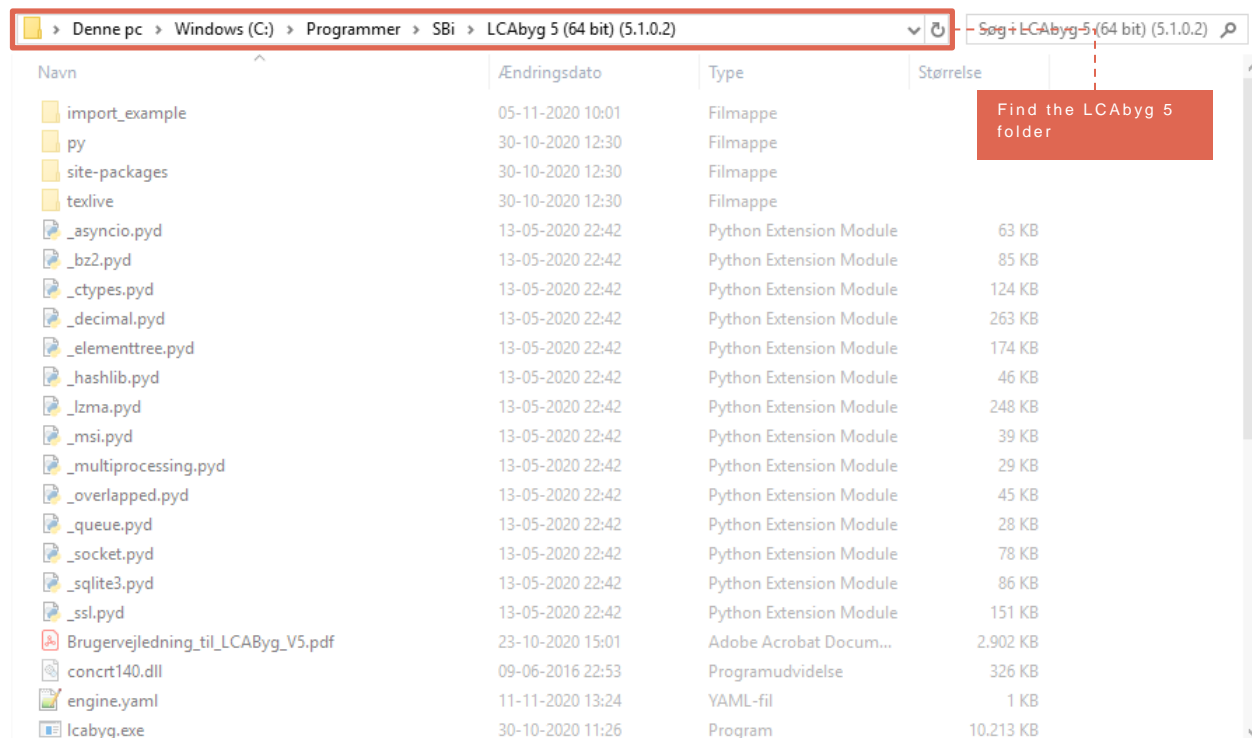
STEP 3 - CREATE A NEW PROJECT TEMPLATE

You can create a new project template like the empty project and case1. By creating a new project template you can change e.g. the defaults for a project in LCAbyg, which can be beneficial if you use the same template for a project over and over.

In this example, the project template named Import Example has been created, which can be used and modified by the user. The project template already contains a number of json files that can be modified and added to.

Step 3.1 - Find the LCAByg 5 folder

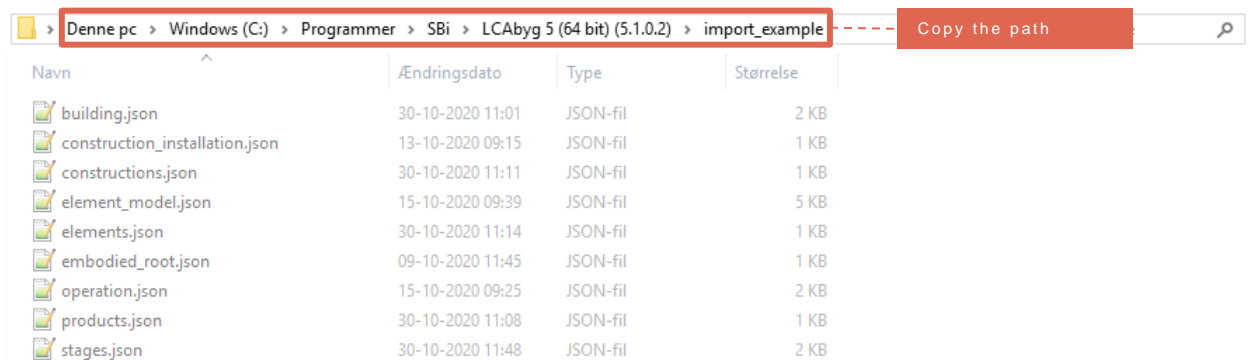
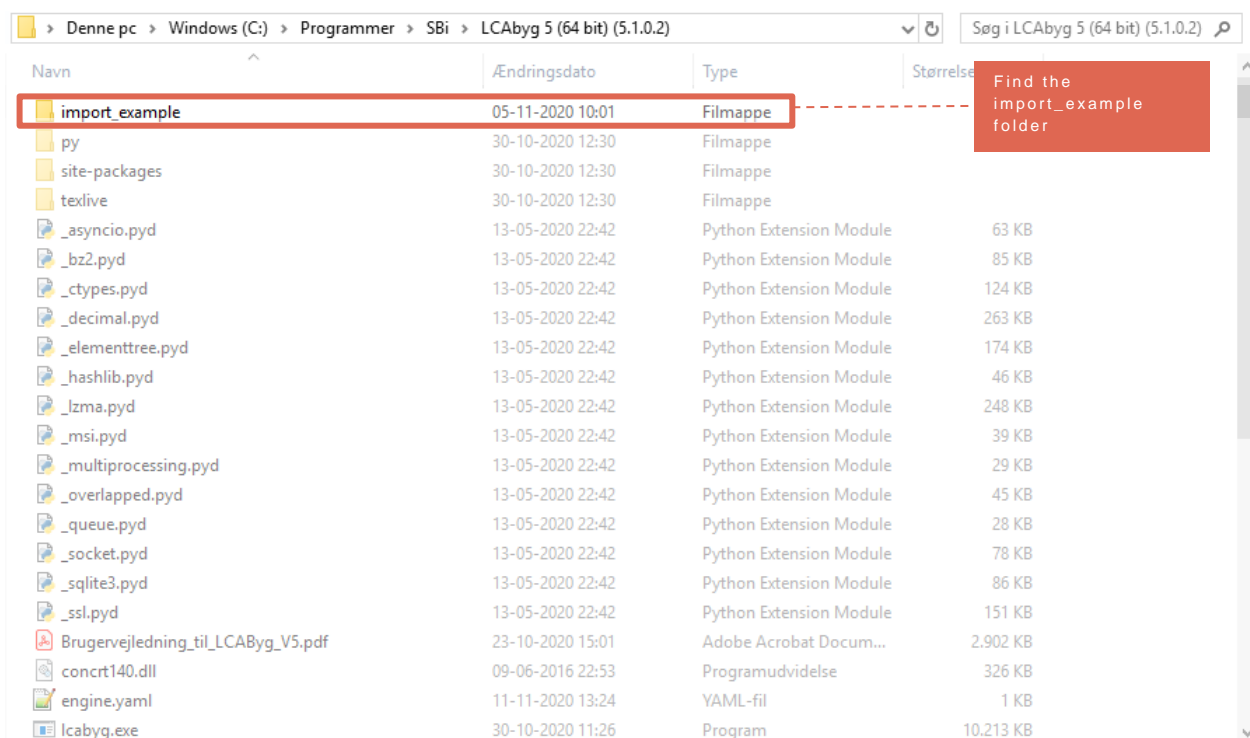
The first step in the creation of a new project template is to find the LCAByg 5 folder. When downloading LCAByg you chose the location of the folder. In the vast majority of cases, the folder should have been located in C:\Program Files\SBi\LCAByg 5 (64 bit) (5.1.0.2)



Step 3.2 – Find the project template folder

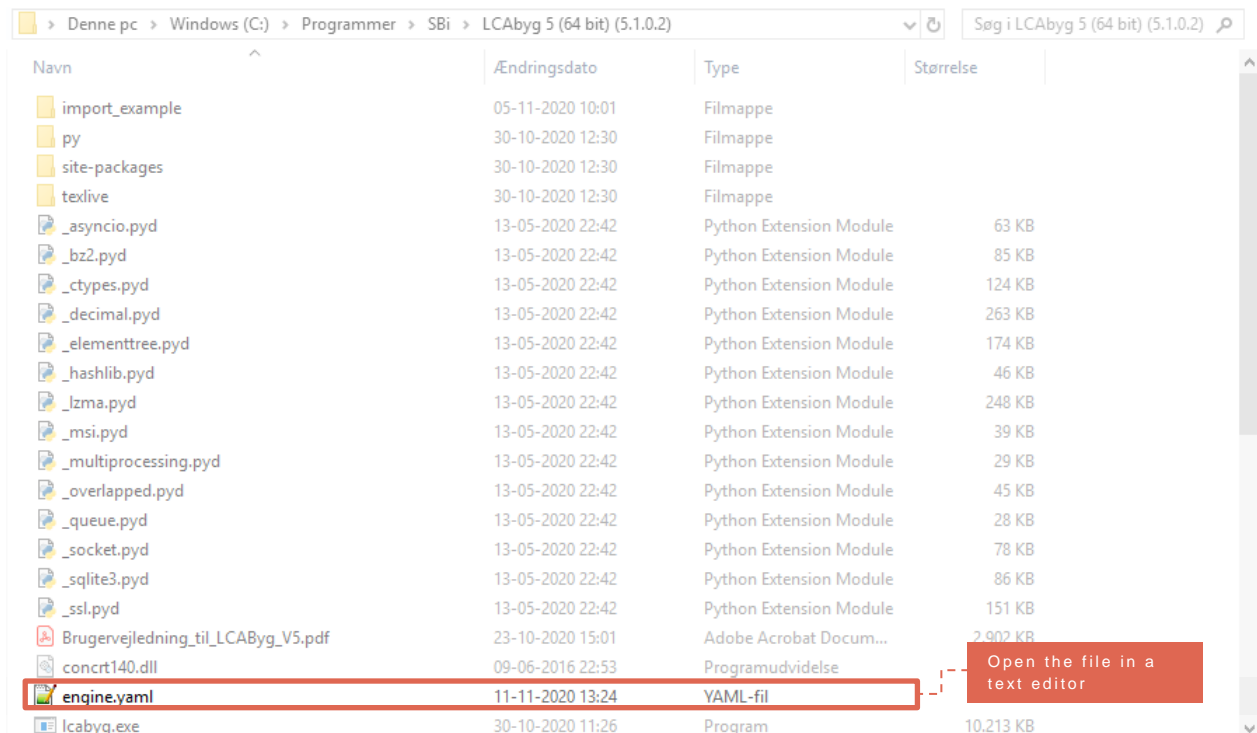
After locating the LCAByg 5 folder you must find the project template folder named import_example. **Open** the **import_example** folder and **copy** the **folder path**. Inset the copied folder path in your Notepad or a Word document as you will need this path later in Step 3.4.

Note: In the LCAByg 5 folder you can create your project template by creating a folder like import_example, name the folder and locate it in the LCAByg 5 folder. We recommend working with the import_example, to begin with.



Step 3.3 - Open engine.yaml file in a text editor

Open the file engine.yaml in a text editor.



Note: you do not have administrator rights please copy the engine.yaml file for later use.

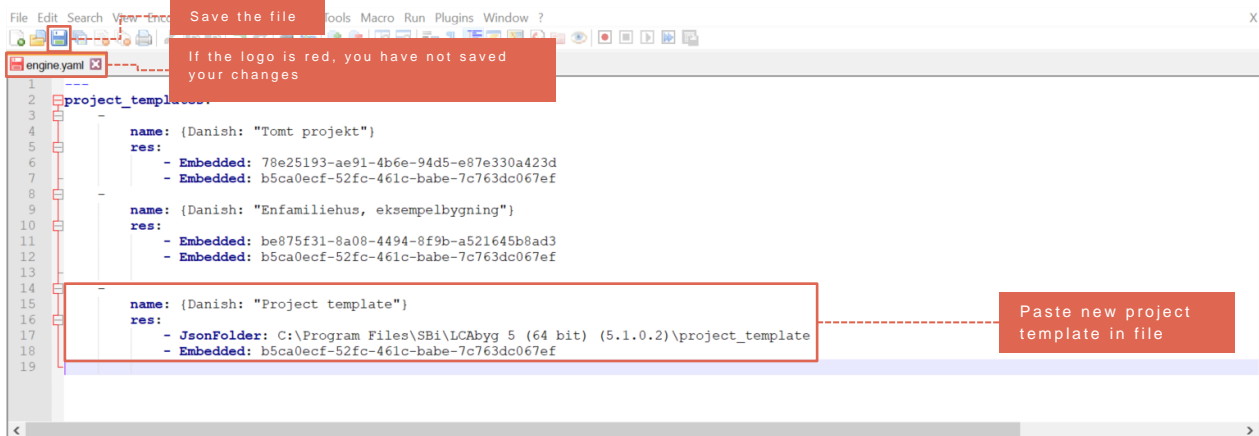
Step 3.4 - Copy and paste an existing template in engine.yaml file

After opening the engine.yaml file you have to create a new project template. Copy the following text in the box and place it after the existing project templates in the engine.yaml file.

```
-  
  name: {Danish: "Project template"}  
  res:  
    - JsonFolder: C:\Program Files\SBi\LCAByg 5 (64 bit) (5.1.0.2)\import_example  
    - Embedded: b5ca0ecf-52fc-461c-babe-7c763dc067ef
```

If you want to change the name of the project template, enter a new name in the quotation marks (" "). The folder path created in Step 3.2 should be placed in the text field highlighted with yellow.

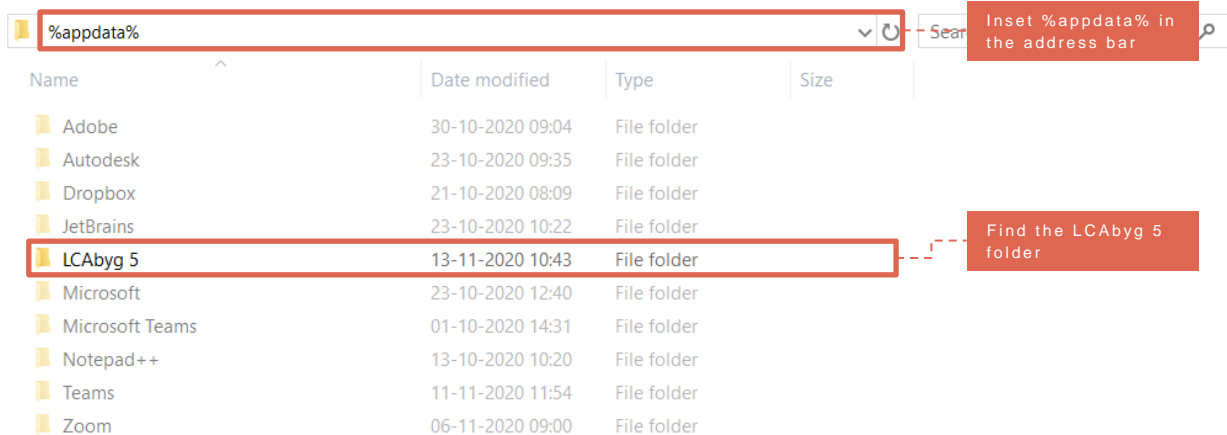
Remember to save the engine.yaml file after placing and editing the new project template. If the "save logo" turns red, you have not saved the changes in the file. Save the file by clicking on the save logo in the menu.



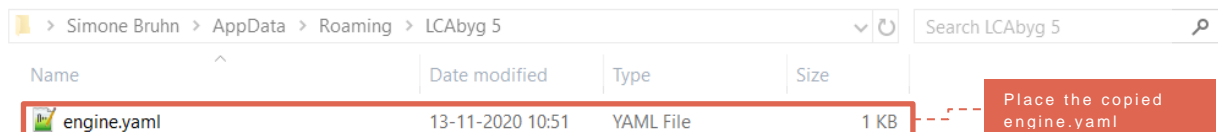
Step 3.5 – No administrator rights

If you do not have administrator rights please follow the following steps.

Copy the engine.yaml file from LCAByg 5 folder in which you found in Step 3.1. In your file explorer, write **%appdata%** in address bar. The folder named LCAByg 5 (not the same folder as in Step 3.1) will appear.



Open the LCAByg 5 folder and place the copied engine.yaml file in the folder. Open the engine.yaml file and follow the steps from Step 3.4.



Step 3.5 - Check if the new project template appear in LCAByg

Check if the new project template appear in LCAByg by opening the program as usual. Click on the “**Opret nyt projekt**” logo in the menu in the upper left corner. Click on “**Ny**” and then check if your new project template appears.

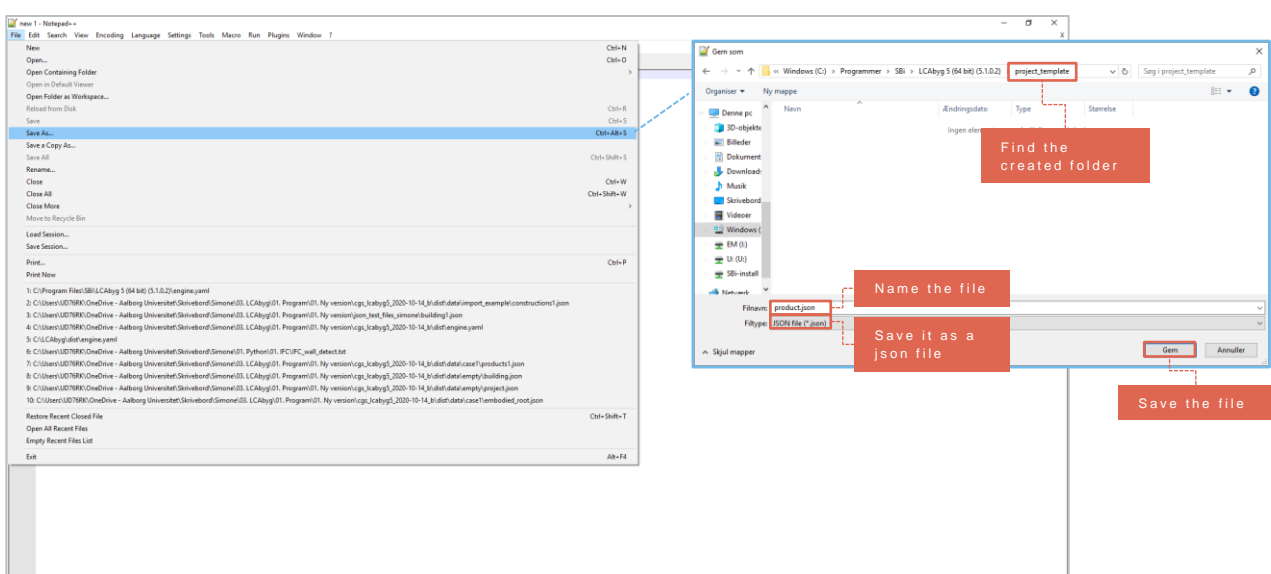
Note: If you have created your own project template and not used the import_example an error will appear and tell that the program is missing a node.

Note: You can also open the file named lcabyg_debug.exe to check if the new project template appears. By opening the debug a terminal of the warnings and errors will appear, and it will be easier to navigate in the warnings and errors.

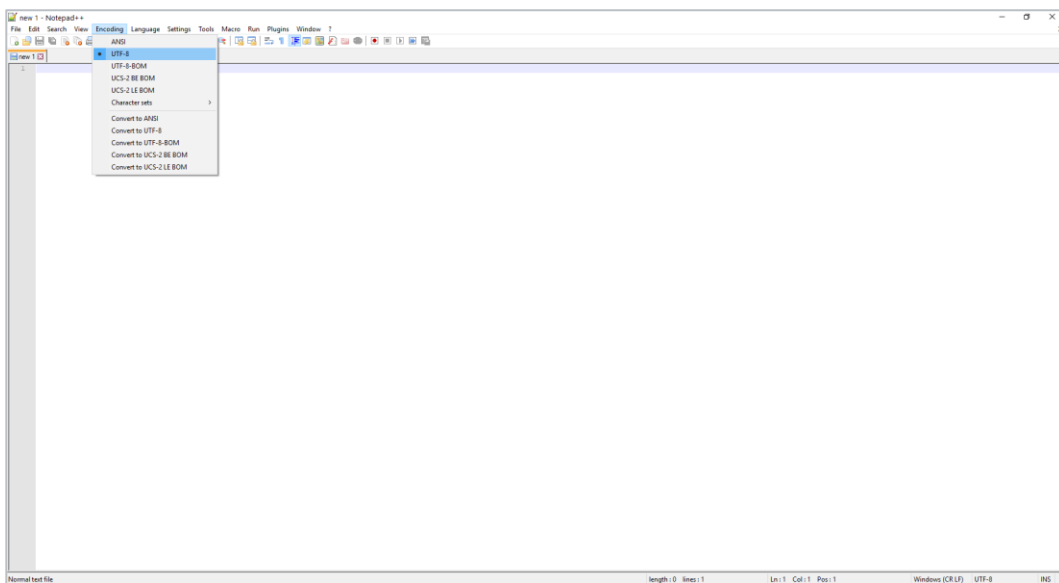
STEP 4 - CREATE NEW JSON FILES

Step 4.1 - Set up a json file

Start by **opening your text editor**. Click on the **file** and then **Save As. Name** you file and **save** your file as a **JSON file (*.json)** in the **folder** named **import_example** which is located in the LCAbyg 5 folder (see Step 3.2). If you have created a new project template folder, you must locate your json files



Note: Remember to set the **Encoding** default to **UTF-8**. You will find the encoding setting in the top bar.



Step 4.2 - Edge requirements

When creating a new library consisting of constructions and products, there is no requirement with Edges in general. In some cases, however, Edges are required e.g. when creating new Stages or/and Elements, see Figure 12.

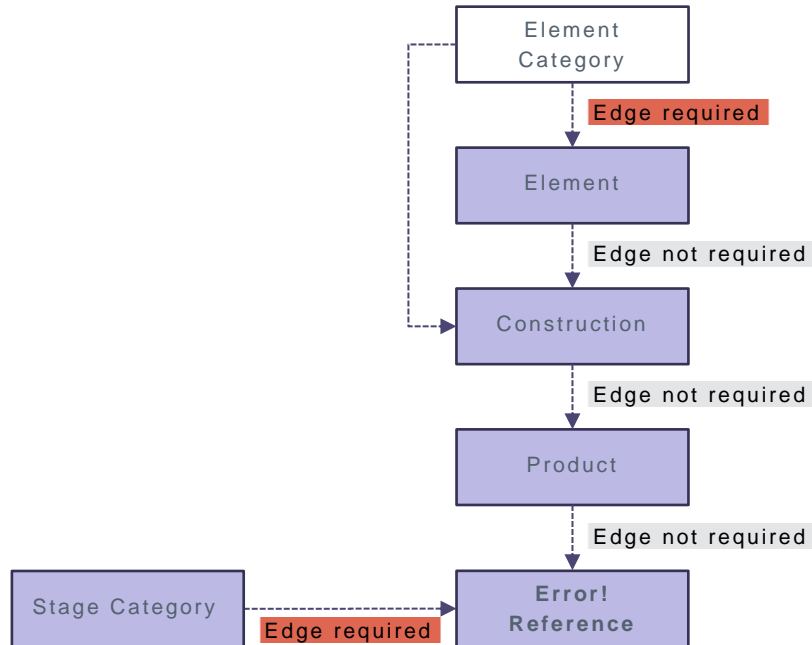


Figure 12 Edges required

When creating an Edges between e.g. a Stage Category and a new Stage or an Element Category and a new Element, you must use the Stage Categories and Element Categories ID's. These ID's can be found in LCAbyg's Library in the gen_dk folder located in the LCAbyg 5 folder.

Sometimes it can be an advantage to create an edge between two nodes, and even if it is not that requirement. An example could be that you want to create a project template where a specific construction always contain a specific building product.

If you want to connect new Nodes or connect a new Node with an existing Node from the gen_dk library a new Edge must be created, see Figure 13.

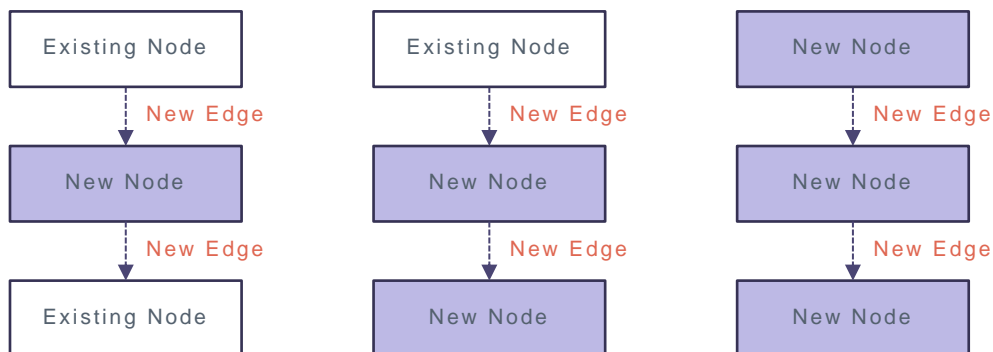


Figure 13 Examples on connections between Nodes and Edges

Step 4.3 - Find ID's and names

The LCAByg library (gen_dk) can be found in the folder gen_dk located in the LCAByg 5 folder. The folder contains all the elements', constructions', building products' etc. IDs and names in which can be found in LCAByg's library. The ID's are used when making an Edge between an existing Node and a new Node.

All the constructions' IDs and names are located in the folder gen_dk in the excel file named **constructions**.

All the products' IDs and names are located in the folder gen_dk in the excel file named **products**.

All the stages' IDs and names are located in the folder gen_dk in the excel file named **stages** etc.

Step 4.4 - The creation of new Nodes and Edges

In the following chapters a review of how to make and set up new nodes for constructions, the product etc. as well as how to create edges between them will be made.

Example	Comment
<pre>{ { "Node": { "Construction": { "id": "Inset_unique_construction_ID_number", "name": { "Danish": "Inset_unique_construction_name" }, "unit": "M2", "source": "User", "comment": "Write_a_comment_if_needed", "layer": 1, "locked": true } } } }</pre>	<pre>Begin dictionary Begin node Begin new construction Create and inset unique ID Create and inset a unique name Set the unit (kg, m, m2, m3 and stk) Inset the source Inset a comment* Inset the layer (1,2 or 3)** End new construction End node End dictionary</pre>

Template for json file

Explanation of the code

Figure 14 Setup example for template for Nodes and Edges

4.1.1 CREATING A NEW ELEMENT (BYGNINGSDEL)

Start to create a new json file and save it in the import_example.

Before creating the node, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new node can be created in the json file.

In the following table, an example of how to create a new element node is explained. You can use the example as a template when creating a new element in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Element Node	Comment
[
{	Begin dictionary
"Node": {	Begin node
"Element": {	Begin new element
"id": "Inset_unique_element_ID_number",	Create and inset unique ID
"name": {	Create and inset a unique name. Remember the Danish, English and German must different form each other.
"Danish": "Inset_unique_element_name"	
"English": "Inset_unique_element_name1"	
"Danish": "Inset_unique_element_name2"	
},	
"source": "User",	The source
"comment": "Write_a_comment_if_needed",	Inset a comment
"locked": true	
}	End new element
}	End node
}	End dictionary
]	

Note: An Edge need to be connected between the Element Node and Element Category Node, see Figure 12.



4.1.2 CREATE A NEW EDGE BETWEEN CATEGORY ELEMENT AND ELEMENT (CATEGORYTOELEMENT)

An edge between a new element and a category element must be created. Start by creating a new json file and save it in the import_example folder. Before creating the edge, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new edge can be created in the json file.

In the following table, an example of how to create a new edge between an element and an element category is explained. You can use the example as a template when creating a new edge in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Note: IDs and names for the category elements can be found in folder gen_dk in the excel file named element_categories.

Edge between element category and element	Comment
[
{	Begin dictionary
"Edge": [Begin edge
{	
"CategoryToElement": {	Begin new edge*
"id": "Inset_unique_edge_ID_number",	Create and inset unique ID
"enabled": true	
}	End new edge
},	
"Inset_element_category_ID_number",	Inset exciting element category ID in excel file
"Inset_element_ID_number"	Inset either exciting element ID or new element ID
]	End edge
}	
]	

* Edge names can be found in Table 1.

4.1.4 CREATING A NEW CONSTRUCTION (KONSTRUKTION)

Start to create a new json file and save it in the import_example.

Before creating the node, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new node can be created in the json file.

In the following table, an example of how to create a new construction node is explained. You can use the example as a template when creating a new construction in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Example	Comment
[
{	Begin dictionary
"Node": {	Begin node
"Construction": {	Begin new construction
"id": "Inset_unique_construction_ID_number",	Create and inset unique ID
"name": {	Create and inset a unique
"Danish": "Inset_unique_construction_name"	name
},	
"unit": "M2",	Set the unit (kg, m, m2, m3 and stk)
"source": "User",	The source
"comment": "Write_a_comment_if_needed",	Inset a comment*
"layer": 1,	Inset the layer (1, 2, 3 etc.)**
"locked": true	
}	End new construction
}	End node
}	End dictionary
]	

*e.g. specifications regarding the construction built up

**see the LCabyg Guide at <https://www.lcabyg.dk/download-program> for more info regarding the layers

Note: No Edge need to be connected between the construction Node and the element or/and element category, see Figure 12.



4.1.5 CREATE A NEW EDGE BETWEEN ELEMENT AND CONSTRUCTION (ELEMENTTOCONSTRUCTION)

An edge between a construction and an element can be created, but is not a requirement. Start by creating a new json file and save it in the import_example folder. Before creating the edge, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new edge can be created in the json file.

In the following table, an example of how to create a new edge between a construction and an element is explained. You can use the example as a template when creating a new edge in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Note: IDs and names for the elements can be found in folder gen_dk in the excel file named elements.

Edge between element and construction	Comment
[
{	Begin dictionary
"Edge": [Begin edge
{	
"ElementToConstruction": {	Begin new edge*
"id": "Inset_unique_edge_ID_number",	Create and inset unique ID
"amount": 59.9,	Inset the amount
"enabled": true	
}	End new edge
},	
"Inset_element_ID_number",	Inset either exciting element ID or new element ID
"Inset_construction_ID_number"	Inset either exciting construction ID or new construction ID
]	End edge
}	
]	

* Edge names can be found in Table 1.

4.1.6 CREATING A NEW PRODUCT (BYGGEVARE)

Start to create a new json file and save it in the import_example.

Before creating the node, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new node can be created in the json file.

In the following table, an example of how to create a new product node is explained. You can use the example as a template when creating a new product in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example requires your input.

Example	Comment
[
{	Begin dictionary
"Node": {	Begin node
"Product": {	Begin new product
"id": "Inset_unique_product_ID_number",	Create and inset unique ID
"name": {	Create and inset a unique
"Danish": "Inset_unique_product_name"	name. Remember the
"English": "Inset_unique_product_name1"	Danish, English and
"Danish": "Inset_unique_product_name2"	German must different form
},	each other.
"source": "User",	The source
"comment": "Write_a_comment_if_needed",	Inset a comment
"locked": true	
}	End new product
}	End node
}	End dictionary
]	

4.1.7 CREATE A NEW EDGE BETWEEN CONSTRUCTION AND PRODUCT (CONSTRUCTIONTOPRODUCT)

An edge between a product and construction can be created, but is not a requirement. Start by creating a new json file and save it in the import_example folder. Before creating the edge, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new edge can be created in the json file.

In the following table, an example of how to create a new edge between a product and construction is explained. You can use the example as a template when creating a new edge in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Note: IDs and names for the construction can be found in folder gen_dk in the excel file named constructions.

Example	Comment
[
{	Begin dictionary
"Edge": [Begin edge
{	
"ConstructionToProduct": {	Begin new edge*
"id": "Inset_unique_edge_ID_number",	Create and inset unique ID
"amount": 0.05,	Set the product quantity used per construction unit*
"unit": "KG",	Set the unit
"lifespan": 100,	Inset the lifespan for the product
"demolition": false,	Set if the product is demolished (false/true)**
"enabled": true,	
"delayed_start": 0	Set the delayed start***
}	End new edge
},	
"Refer_construction_ID_number",	Refer either exciting construction ID or new construction ID
"Refer_product_ID_number"	Refer either exciting product ID or new product ID
]	End edge
}	End dictionary
]	

* Edge names can be found in Table 1.

** The demolition takes place either here and now or after a number of years if a delayed start is chosen

*** A delayed start is chosen when the product is demolished and added in the future



4.1.8 CREATE A NEW EDGE BETWEEN PRODUCT AND STAGE (PRODUCTTOSTAGE)

An edge between a product and stage must be created. Start by creating a new json file and save it in the import_example folder. Before creating the edge, a list which contain a dictionary is created, as illustrated in Figure 6. After this step a new edge can be created in the json file.

In the following table, an example of how to create a new edge between a product and stage is explained. You can use the example as a template when creating a new edge in your json file. Copy the text in purple. Please note that text highlighted with yellow in the example, requires your input.

Note: IDs and names for the stage can be found in folder gen_dk in the excel file named stages.

Example	Comment
[Begin dictionary
{	Begin edge
"Edge": [Begin new edge*
{	Create and inset unique ID
"ProductToStage": {	
"id": "Inset_unique_edge_ID_number",	
"enabled": true	End new edge
}	
},	
"Inset_product_ID_number",	Inset either exciting product ID or new product ID
"Inset_stage_ID_number"	Inset exciting stage ID
]	End edge
}	
]	

* Edge names can be found in Table 1.

4.2 CHECK THE CODE

To check if the code is correct, you must do the following steps:

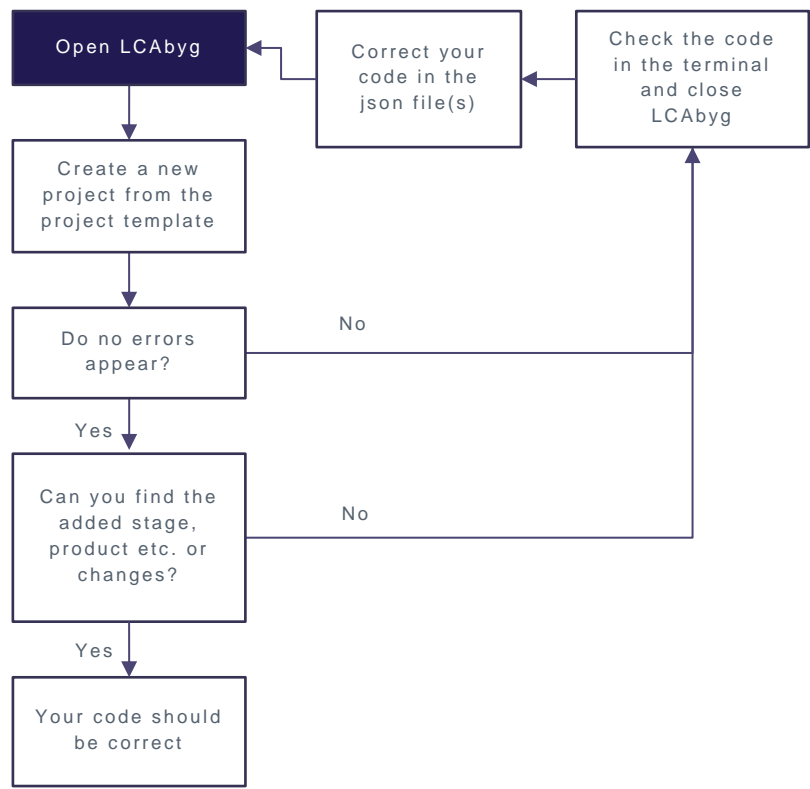



Figure 15 How to check your json code

APPENDIX A

Example on ID form external source (Ökobau)



Collapse all sections

Go back

Close

Process Data set: fire resistant glass-AGC-Pyrobelt; 1 m² area (en) [en](#)

▸ Process information

▸ Modelling and validation

▾ Administrative information

Data entry by

Time stamp (last saved)	2017-02-14T10:13:23.025+01:00
Data set format(s)	<ul style="list-style-type: none">ILCD format 1.1EPD Data Format Extensions
Data entry by	ift Rosenheim

Publication and ownership

UUID	70d35332-9f2d-4f3b-8688-431388263d2d
Data set version	00.00.011
Owner of data set	AGC Glass Europe
Copyright	Yes

▸ Environmental indicators

Inset the text highlighted with yellow, when creating a new stage.