Documents Manager

Documents Manager panel allows managing and customizing documents available in Leon, such as: **Flight Order**, **Trip Sheet**, **General Declaration**, **PAX Information**, **PAX Manifest**, **Flight Brief** & **Charter Agreement**.

The list of pre-defined documents is constantly expanded to cover all documents in Leon. It is also possible to create completely **new custom document** apart from those pre-defined in Leon.
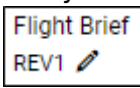
Main page

Documents Manager main screen

The main page shows **3** filtering options: by **document type**, **language** and **publishing status**, as well as **7** columns:

- **Name** - name and the saved versions of the document
- **Type** - type of the document
- **Published** - the status of the documents, whether they have been published, or not. Click on the white dot if you want to publish the saved version - the dot will become green
- **Visible in Crew App** - allows indicating which version(s) of the document will be available to Crew on a flight in the Crew App > tab 'Flight'. The dots indicate the status: 'White' - not available to the Crew; 'Grey' - not applicable - document is not published; 'Green' - available to the crew
- **Language** - the language of the document indicated within the edit in the SETTINGS tab
- **Last modified** - the date of the latest modification.
- **Action** - once the document has been uploaded you can hover the mouse to see available options:

1. - the edition of the document
2. - saving the copy of the modified document
3. - deleting saved document, only available on not published documents
4. - viewing of the document

Filter work immediately after selecting the value. To disable the filter just click the black arrow in the filter bar.

It is possible to manually change the document's version name by hovering a mouse over it and typing a new one -

Adding your logo to documents

Adding your own logo to existing documents is an easy task which should be done at the beginning of using Leon.

All the documents are coded to display a **logo.png** file as a logo by default. To have your own logo displayed on all your documents, follow the steps below:
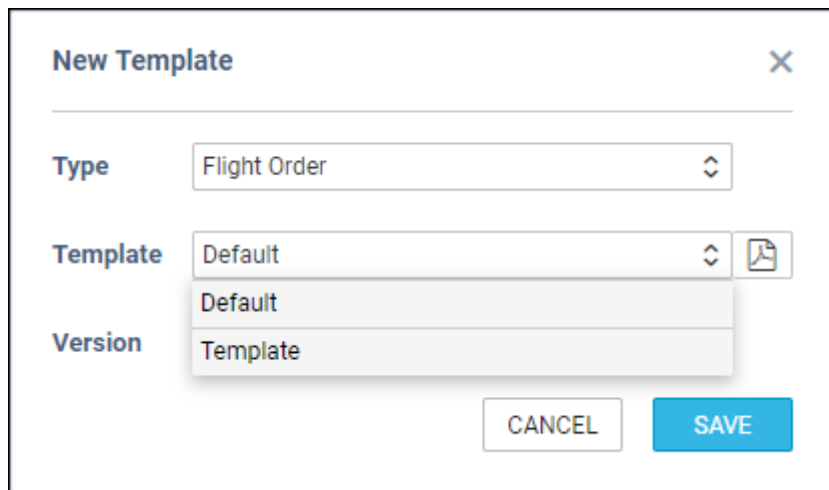
1. Prepare a file with your logo. It must be a PNG file, named **logo.png**.
2. Go to Settings → Documents Manager, and edit any document you find there.
3. Click the ping icon you can find in the top right corner.
4. In the right panel, go to the **FILES** tab.
5. In the **FILES** tab, find a **logo.png** file which can be there, and remove it by clicking the trash bin icon.
6. In the same place, upload the file you prepared in the 1st step.
7. Here you are - your logo should be displayed on all documents which have the default coding.

> The files uploaded to the **FILES** tab are available to all documents in Documents Manager, so you only need to upload your logo image to any of them, and it will be shown in all of them.

Creating documents

In order to create a new document click **NEW TEMPLATE** button. In a pop-up window select the desired document type, then select one of the available templates from which a new document will be created. Some documents have more than just one template prepared by Leon based on operators experiences - choose the most suitable template for your needs. To preview a selected template click on the document icon next to the template field bar. Finally, enter the version name - it will help to recognize the document in the OPS section, if more than one document of the same type is published. Saving data will create a new document based on the selected template.

When uploading a new Flight Order template (use **NEW TEMPLATE** button again) Leon gives a choice between 2 different Flight Order templates.
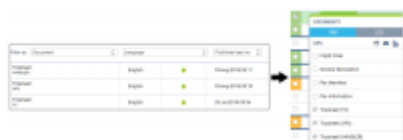
Click on the icon to view which version you prefer.

Publishing documents

Publishing multiple versions of the same document

To publish a document which has been re-arranged you just need to click on the white dot ○ which will turn green 🟢 when a document is successfully published.

Such a published document can be used in a section **OPS** (right-click with the mouse over the trip).

Unpublishing documents works exactly in the same way - the green dot turns white and Leon shows a message in top-right corner: "Template successfully unpublished!".

It is possible to publish **multiple versions** of the same document type. All of them will appear in the main Documents Manager page as well as in OPS documents list and can be recognized by the revision name.

DELETING DOCUMENTS

> **Deleting documents will <u>permanently</u> remove the selected document. <u>Restoring deleted document is not possible</u>. In order to delete a document you need to**
>
> **unpublish it in first place, then click 🗑 icon and confirm the action.**

Customizing documents

Documents Manager - Body of the document

The most important feature of the Documents manager is the ability to customize documents for

operator's **specific needs**. To open the panel dedicated for customizing documents click icon in the document's context menu. Document editor panel is also opened after copying the existing document. Document editor consists of **three panels**: text editor, output document preview panel and right settings panel.

From the Documents editor you can perform following operations:

- prepare, preview and save changes to a document.
- make a copy of a document.
- view the data tree available to be used in a document.
- manage files available to be used in a document.
- change the page settings for a current document.
- return to the documents list.

Below are some websites that might help you work with the 'Document Manager':

- **Twig - document template engine** - https://twig.symfony.com/doc/2.x/templates.html
- **HTML and CSS tutorial** - https://learn.shayhowe.com/html-css/
- **HTML Tables** - https://learn.shayhowe.com/html-css/organizing-data-with-tables/
- **CSS basics** - https://learn.shayhowe.com/html-css/getting-to-know-css/

> **If you don't have IT experience, we advise you contact your IT team to help you in modifying documents**

Copying document

System always stores only the **current version of the document**. This means that a document cannot be restored to the state before applying changes. Because of that it is recommended that preparing the new version of a document should be made on the document copy. When the new version of a document is ready you just need to publish the new version and unpublish the old version.

To make a copy of document click "Save as Copy" from the buttons bar. In a pop-up form enter a new revision name and save. Manager will create and save a copy of the current document. User will be redirected to editing document copy.

File uploaded to the Documents manager can be used in every document created in the Documents

manager. To upload a file open the right panel on FILES tab. Click upload files button and select image file you want to upload. **Only jpg, jpeg and png files are supported**. After selecting the file it will be automatically uploaded into the Documents manager and displayed on the list of available files. Name of the files are used to insert images into documents. Try to keep images filenames short and representative. To preview an image hover mouse over file name. Small tooltip will show image contents. In order to delete the file select trash icon next to file name.

Page settings

Page settings allow configuration of the page **margins** and the page **orientation**. Some documents require more space for the header and the footer, the other ones are printed in landscape instead of portrait view.

In order to change page settings select **SETTINGS** tab on the right panel in document editor. Make required changes and save them with SAVE button at the bottom of the SETTINGS tab. Saving settings reloads the document preview to check applied settings. Page settings are applied only for edited document.

Process of customizing documents

Documents manager is a tool combining many **technologies** used by developers to create and update web applications. A particular document is created by converting **HTML** document to **PDF** file. HTML document is filled in with selected trip data by **Twig** template and styled by **CSS** (with optional images resources). Full understanding of all used technologies is not required to customize document. User without any programming knowledge can easily change static fragments of documents and experiment with other constructions peeking existing templates. For better understanding the power of documents manager it is recommended to know at least basics of technologies used in manager.

Each document consists of **3** parts: **header**, **body** and **footer**. Parts are analogical for real documents that can have header and footer copied on every page of a document. Every part has its own HTML editor and can be managed separately. There is a single CSS for all document parts.

Adding aircraft pictures to documents

If you want to add pictures of your aircraft on a particular document, you need to first **upload pictures** to particular tail in a section SALES > FLEET (edit the aircraft and use 'EDIT PICTURES' button), name them and SAVE (click also SAVE at the bottom of the page). Once it's done, edit a particular document and use below HTML and CSS entries in Documents Manager panel.

The example below shows how to add 2 pictures of the aircraft: the outside look and the inside one, plus a command that pictures should always appear on the next page and be separated from each other.

BODY

```
1.  <div class="pictures">
2.    <div>
3.      <img
    src="{{PAXInformation.flights|first.aircraft.pictures|getAircraftPictur
    e("aircraft").url}}"/>
4.      <img
    src="{{PAXInformation.flights|first.aircraft.pictures|getAircraftPictur
    e("inside").url}}"/>
5.    </div>
6.  </div>
```

CSS

```
1.  .pictures {
2.    page-break-before: always;
3.    line-height: 1cm;
4.  }
```

Attaching extra pages to documents

It is possible to attach **extra pdfs** to existing documents in Leon. If, for example, Flight Order needs to have attached an extra page for pilots with towing instructions, you can attach it by uploading pdf file into Documents Manager panel (edit the document and use right-hand filter, tab FILES) and by adding below HTML code at the bottom of the 'BODY' HTML code:

```
{% pdf 'towing_instructions.pdf' %}
```

where 'towing_instructions' is the name of the pdf document.

Attaching pdf documents from the checklist items to Crew Tripsheet

It is possible to attach pdf files uploaded to some of the elements of the checklist to the Crew Tripsheet document.

This can be achieved by using the function `{{ showFiles(files) }}`.

This is only possible in the Crew tripsheet for the elements that contain 'file' link in the 'Available data', e.g. tripsheet.activities.checklist.catering.files

An example of how to embed the code in the tripsheet can be seen below, where the 'leg' element pulls out the details from single legs and 'trip' element pulls out the details from the whole trip.

```
1. {% set files  = [] %}
2.   {% for leg in tripsheet.activities %}
3.    {% if leg.checklist.catering and leg.checklist.catering.files %}
4.         {% set files = files|merge(leg.checklist.catering.files) %}
5.   {% endif %}
6. {% endfor %}
7. {% for trip in tripsheet.trips %}
8.       {% if trip.checklistTrip.CATBriefsRequired and
   trip.checklistTrip.CATBriefsRequired.files %}
9.           {% set files =
   files|merge(trip.checklistTrip.CATBriefsRequired.files) %}
10.   {% endif %}
11. {% endfor %}
12. {{ showFiles(files) }}
```

The PDF file(s) will be embedded at the **end of the document**.

Showing a map in documents


Showing a map in documents

It is also possible to **show a map** in particular documents. In order to do that you need to insert the below code:

```
1. <map route={{ trip.flights|getRouteForMap}} width="800" height="500"
   line="purple" />
```

You can define **width** and **height** as you wish, as well as the **colour of lines** on the map (see screenshot).

The parameters set by default: "c_width=2" and "markers_icon_url" which have a format of "red_markers.png" can also be changed. Markers are visible by default, if you want to remove them, set a parameter as "none". They are red circles that look like:
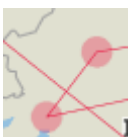
Adding header/footer as a picture

To add either a **header** or a **footer** as a picture you need to use appropriate tabs in Documents Manager panel.
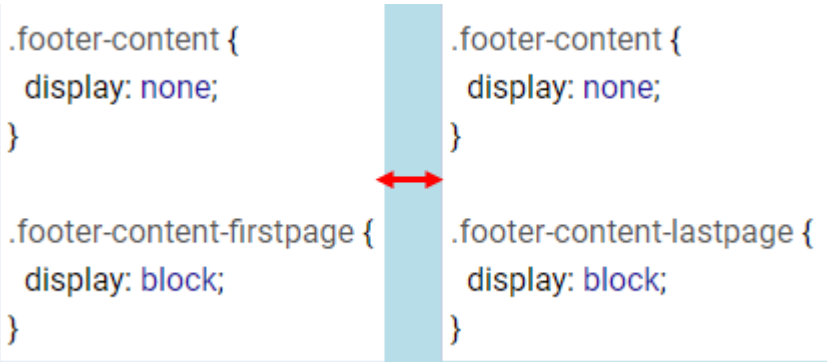
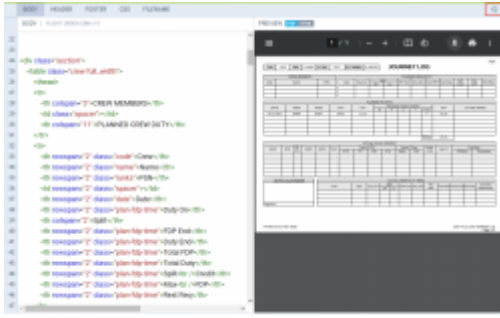First of all a file should be uploaded to the right-hand filter, tab **FILES**.
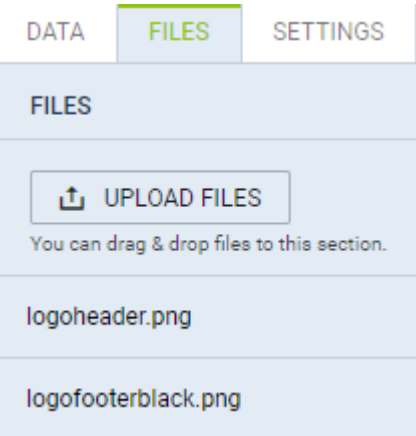
Once it's done, a **code** needs to be inserted in a tab HEADER:

Then, in a **CSS** tab you can set up, that i.e. the footer should appear only on the 1st page, or that the footer should appear only at the bottom of the last page:
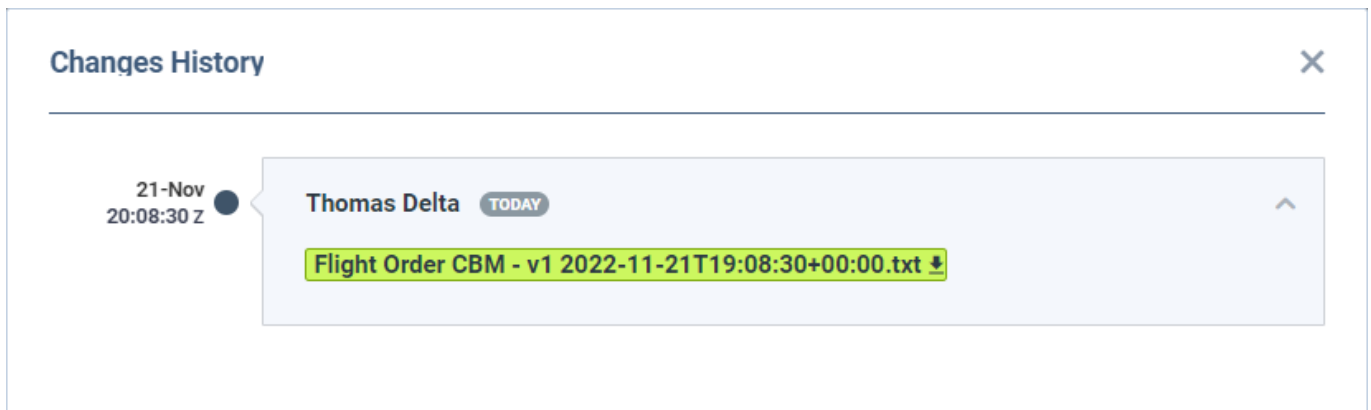

Changes history

Changes history icon

It is also possible to see the history of changes of the document as well as to preview the past versions of the code.

To access the history of changes, simply click on the icon available just above the document preview section (as per the screenshot on the right).

Len will display the changes history information such as time and date (in UTC) and the name of the user who made changes (as per below):

Clicking on the icon within the changes history link allows saving the selected version of the document (code) to a text file.

This will allow retrieving any past version of the document by pasting the code from the document to the relevant section of the Document Manager.

HTML structure

HTML is a markup language that allows defining structure of web pages. Pages written in HTML can be further styled by CSS. In Documents manager user only writes HTML placed inside <body> tags. HTML headers are added automatically outside of the editor. Be aware that header, body and footer are treated as separated HTML documents but they share single CSS file.

In HTML every element, like static text (text that won't be changed by dynamic trip data) should be inside HTML tags. For example text can be wrapped in universal <div> tag.

<div>

basic text

</div>

This text will be displayed in a document with a default font colour and the size. That and more can be changed using CSS.

With HTML you can insert previously uploaded **images** to the document. In order to add an image insert tag <img> with the document file name as source attribute. For example, if you want to add aircraft.png image (already uploaded to manager) to a document put following HTML: <img src="aircraft.png"> This tag will show image in place of document where tag was inserted. We encourage you to familiarize yourself with most popular HTML constructions like list and table.

For more information regarding HTML see:

HTML and CSS tutorial

HTML Tables

CSS structure

CSS is used to add **style** to document's structure. CSS capabilities can be compared to advanced text editor. For example, user can change: elements of the **font**, **font size**, **colour** of the text or the **background**, can add **margins**, **padding** and **borders**, can align **content**. There are a lot more CSS options to apply to a document's elements.

To apply style to document elements you can use general **tags** like **<div>** or <table> or you can name elements in the document adding **identifiers** (id) and **classes** to HTML attributes. It is possible to apply style only to named elements in a document.

Below is an example with a simple usage of general and specific tags:

```
div {
        margin-top: 20px;
}


.schedule_table {
        color: red;
}
```

The above style adds a top margin to all div elements and sets red text colour to all elements with class named "schedule_table".

The most important CSS feature is that **styles can override each other** from general to specific.

This way you can set standards for all elements in a document (like font size) and then override and extend this style in specific elements across the document.

For more information regarding CSS see:

[CSS Basics](#)

Using Twig

Twig is a template engine that can insert **dynamic data** from LEON into document's HTML structure. Before using Twig you should familiarize yourself with data structure of a document you are customizing.

Click on the **DATA** tab in the right panel. System displays data structure for a selected trip (it can be default or selected by an user). Now explore data structure by **expanding nodes** clicking on them. There are **3** types of nodes:

- objects with named fields.
- array structures where fields are numbered indexes (starting from 0).
- scalars (words, numbers, true/false values) – not expandable further.

Knowing the structure is important because placing the data tag with Twig requires **setting path** to the desired element. Path should be analyzed carefully taking into account optional fields and arrays. If on the path of data you want to show is array you should iterate over this array as it can have different number of elements or have no elements.

Twig has **2** kind of tags: **data tags** and **control tags**. Data tags allow displaying data. For example simple tag to show a trip number from flight brief structure looks like this:

```
{{ trip.tripNo }}
```

If you put this tag in a document's template then, after the preview, in this place Manager will show a trip number from a selected trip.

Control tags give an advanced **control** to an user over the **template**. Most common control tags are **conditions** and **loops**. Conditions allow to show certain parts of templates based on condition. Example of checking if trip client is set is:

```
{% if trip.client %} - start condition block with condition check
      <div class="client_name">
                {{trip.client.name}} – shows trip client name if trip has client
      </div>
{% else %} - start condition alternative block (alternative block is optional)
      <div>Trip has no client</div> - shows information if trip has no client
{% endif %} - end condition
```

The above code will show trip client name only if the trip have a client selected in Leon, otherwise

document will show an information that the trip has no client. Usage of condition alternative is optional, you can add alternative block only when needed.

Loops allow iterating over arrays. For example, iterating over flights (available in path trip.flights) can look like this:

```
{% for flights in trip.flights %} - start of loop block, means for every flight in trip.flight array show as follows
        <div>{{flight.flightNo}}</div> - shows flight number for every flight in array
{% endfor %} - end of loop block
```

The above construction will add to a document <div> tag with the flight number for every flight in a trip. If there will be no flights - no <div> will be generated. As you can see, the structure named 'flight' represents every iterated value from trip.flights array.

TWIG FILTERS

Data placed in a document by Twig tags can be modified using **Twig filters**. There are many filters at your disposal, provided by Twig and added by LEON team for special cases. Using the filter is simple. For example, adding date filter to format **UNIX timestamp** (number of seconds from 1.01.1970) looks as follows:

```
{{ flight.startTime|date('d-m-Y') }}
```

Date filter takes the date format as a parameter. Filters can take any number of parameters (including none) depending on their specification. It is possible to chain filters, next filter works on an input provided by the output of the previous filter. See the full list of available filters in Twig specification below.

CUSTOM TWIG FILTERS

LEON team prepared **custom filters** to simplify operations on the available data. Be aware that custom filter are dedicated to specific data types and **cannot** be used on other data. Here is a list of all custom filters:

* defaultLicenseNumber – returns pilot license number from array of all license objects. If a pilot has no license in array - an empty word is returned and nothing is shown on document.

example usage

```
{{ crewMember.licenses|defaultLicenseNumber }}
```

example output: 123456789

- exchangeOrderInCrewsByPositionName – changes the order of crew positions array replacing crew member order from one position to another.

example usage

```
{{ flight.uniqueCrewList|exchangeOrderInCrewsByPositionName('FO', 'CPT2') }}
```

example output: returns the same crew members list with replaced order of FO and CPT2 positions.

- formatTimePeriod – formats time period in seconds to hours and minutes format HH(H):mm. Filter is most commonly used to show time difference between two timestamps or aggregated trip times.

example usage

```
{{ flightTimeTotal|formatTimePeriod }}
```

example output: 124:59

- firstFlightWithAircraft – returns first flight in array that have specific aircraft selected. In special cases some flights have no aircraft selected. Using this filter you can get first flight with an aircraft for further processing.

example usage

```
{{ trip.flights|firstFlightWithAircraft.flightNo }}
```

example output: flight number of first flight with aircraft.

- firstProperFlightActivity – return first flight activity (not positionings).

example usage

```
{{ fitstProperActivity = trip.flights|firstProperFlightActivity.startTime|date('d-m-Y') }}
```

example output: start date of first flight not positioning.

example usage

{% if FDPs|sameFDPs %}

example output: checks if all FDP elements are the same (duty start, duty end, duty lenght, next EOBT).

PAGE NUMBERING

Sometime there is a need for **page numeration**. Twig allows page numbering in header and footers. In order to add the page number use following control tags in header or footer:

```
{% pagenumber %} - tag is replaced for current page number
{% pagetotal %} - tag is replaces by total page number in whole document
```

For more information regarding Twig see:

Twig documentation

Twig filters

From:

https://wiki-draft.leonsoftware.com/ - **Leonsoftware Wiki**

Permanent link:

https://wiki-draft.leonsoftware.com/leon/documents-manager

Last update: **2023/11/15 11:53**