Doc Fields in detail

${PROJECT}

The **${PROJECT}** field is used to output a project property's value or based on a template written for the project.

Here is an example of **${PROJECT}**:

```
${PROJECT, PROPERTY=name}
```

Here is the sample output:

```
MyProject
```

This is the syntax of a **${PROJECT}** field:

```
${PROJECT,     template_name | PROPERTY=property_name  }
```

This is a description of the various parts of a **${PROJECT}** field:

- PROJECT is to indicate that this is a **${PROJECT}** field.
- **template_name** | PROPERTY=***property_name*** – The type of content to be extracted from the project and printed on the document.
    - ***template_name*** – Output content from the project based on the template ***template_name*** written for the project. For example, if you have a template *AllClassDiagrams* written for project, by specifying *AllClassDiagrams* as ***template_name***, Doc. Composer will output content by following *AllClassDiagrams*.
    - PROPERTY=***property_name*** – Output a **specific** property (e.g. name) for the project. If you want to output multiple properties, try write a template and make reference to it by providing its name here.

${DIAGRAM}

The **${DIAGRAM}** field is used to query diagram(s) from a project (or a specific place in a project), and to output content from the diagrams querying.

Here is an example of **${DIAGRAM}**:

```
${DIAGRAM, "List of Use Case Diagrams", "UseCaseDiagram", LoopInProjec
t, PROPERTY=name}
```

Here is the sample output:

```
Use Case Diagram1, Use Case Diagram2, Use Case Diagram3
```

This is the syntax of a **${DIAGRAM}** field:

```
${DIAGRAM,     field_name,     ["diagram_type{,
diagram_type...}", ]    [SortBy="property{,property
...}", ]
One | Any | LoopInProjec
t | LoopInElement,    template_name | PROPERTY=
property_name | ICON | IMAGE  }
```

This is a description of the various parts of a **${DIAGRAM}** field:

- DIAGRAM is to indicate that this is a **${DIAGRAM}** field.
- ***field_name*** is a short description of the field (e.g. "List of Use Case Diagrams"). In Doc. Composer, you can see the fields placed in an imported Doc Base. The fields are represented by the ***field_name*** typed here. ***field_name*** must be unique within a Doc Base. If there are two or more fields having the same field name, content may be produced wrongly.
- ***diagram_type*** indicates the type(s) of diagram that you want to query (e.g. "UseCaseDiagram"). If you want to query all types of diagram in a project, skip this parameter. If you want to query multiple types of diagram, enter their types respectively, separated by comma (e.g. "ClassDiagram,UseCaseDiagram"). Click here for the proper diagram types to use.
- SortBy is an optional part that supports the sorting of diagrams retrieved, based on the property or properties specified. If you want diagrams not to be sorted, use *SortBy=NoSort*.
- One | Any | LoopInProject | LoopInElement indicates the source from which to query diagrams.
    - One – Query a **specific** diagram in project. This option is often used when your document, or part of the document is written around a specific diagram. If you choose One here, you will have to select in Doc. Composer the diagram to query.
    - Any – Query a number of diagrams in project. If you choose Any here, you will have to select in Doc. Composer the diagram to query.
    - LoopInProject – Query all the diagrams in project.
    - LoopInElement – Query the sub-diagram(s) of a specific model element. If you choose LoopInElement here, you will have to select in Doc. Composer the model element from which to query sub-diagrams.
- **template_name** | PROPERTY=***property_name*** | ICON | IMAGE – The type of content

to be extracted from the querying diagrams and printed on the document.

- ***template_name*** – Output content from each of the querying diagrams, based on the template ***template_name*** written for the type of the querying diagram. For example, if you have a template *AllUseCases* written for Use Case Diagram, by specifying *AllUseCases* as ***template_name***, Doc. Composer will output content for each of the Use Case Diagrams by following *AllUseCases*.
- PROPERTY=***property_name*** – Output a **specific** property (e.g. description) for each of the querying diagrams. If you want to output multiple properties, try write a template and make reference to it by providing its name here.
- ICON – Output the icon for each of the querying diagrams.
- IMAGE – Output the diagram image for each of the querying diagrams.

${ELEMENT}

The **${ELEMENT}** field is used to query model element(s) or diagram element(s) from a project (or a specific place in a project), and to output content from the elements querying.

Here is an example of **${ELEMENT}**:

```
${ELEMENT, "List of Use Cases", UseCase, LoopInProject, PROPERTY=name}
```

Here is the sample output:

```
Use Case1, Use Case2
```

This is the syntax of a **${ELEMENT}** field:

```
${ELEMENT,     field_name,     ["element_type{,
element_type...}", ]    [SortBy="property{,property
...}", ]    One | Any | LoopInProject | LoopInElement | LoopInDiagram,
    template_name | PROPERTY=property_name | ICON  }
```

This is a description of the various parts of a **${ELEMENT}** field:

- ELEMENT is to indicate that this is a **${ELEMENT}** field.
- ***field_name*** is a short description of the field (e.g. "List of Use Cases"). In Doc. Composer, you can see the fields placed in an imported Doc Base. The fields are represented by the ***field_name*** typed here. ***field_name*** must be unique within a Doc Base. If there are two or more fields having the same field name, content may be produced wrongly.
- ***element_type*** indicates the type(s) of model/diagram element that you want to query (e.g. "UseCase"). If you want to query all types of model element in a project, skip this

parameter. If you want to query multiple types of model element, enter their types respectively, separated by comma (e.g. "Actor,UseCase").

- SortBy is an optional part that supports the sorting of elements retrieved, based on the property or properties specified. If you want elements not to be sorted, use *SortBy=NoSort*.
- One | Any | LoopInProject | LoopInElement | LoopInDiagram indicates the source from which to query elements.
    - One – Query a **specific** model element in project. This option is often used when your document, or part of the document is written around a specific model element. If you choose One here, you will have to select in Doc. Composer the model element to query.
    - Any – Query a number of model elements in project. If you choose Any here, you will have to select in Doc. Composer the model element to query.
    - LoopInProject – Query all the model elements in project.
    - LoopInElement – Query the child elements of a specific model element. If you choose LoopInElement here, you will have to select in Doc. Composer the model element from which to query child elements.
    - LoopInDiagram – Query the diagram elements from a specific diagram. If you choose LoopInDiagram here, you will have to select in Doc. Composer the diagram from which to query diagram elements.
- ***template_name*** | PROPERTY=***property_name*** | ICON | IMAGE – The type of content to be extracted from the querying elements and printed on the document.
    - ***template_name*** – Output content from each of the querying elements, based on the template ***template_name*** written for the type of the querying element. For example, if you have a template *UseCaseInfo* written for Use Case, by specifying *UseCaseInfo* as ***template_name***, Doc. Composer will output content for each of the Use Cases by following *UseCaseInfo*.
    - PROPERTY=***property_name*** – Output a **specific** property (e.g. description) for each of the querying elements. If you want to output multiple properties, try write a template and make reference to it by providing its name here.
    - ICON – Output the icon for each of the querying elements.

${ICON}

When you are working with a table, you can place the **${ICON}** field in a table cell to let Doc. Composer replace it with the icon image of the querying diagram or element.

Note that **${ICON}** can only be used in a table cell.

Here is an example of **${ICON}**:

```
${ICON}
```

Here is the sample output:

This is the syntax of a **${ICON}** field:

```
${ICON}
```

${IMAGE}

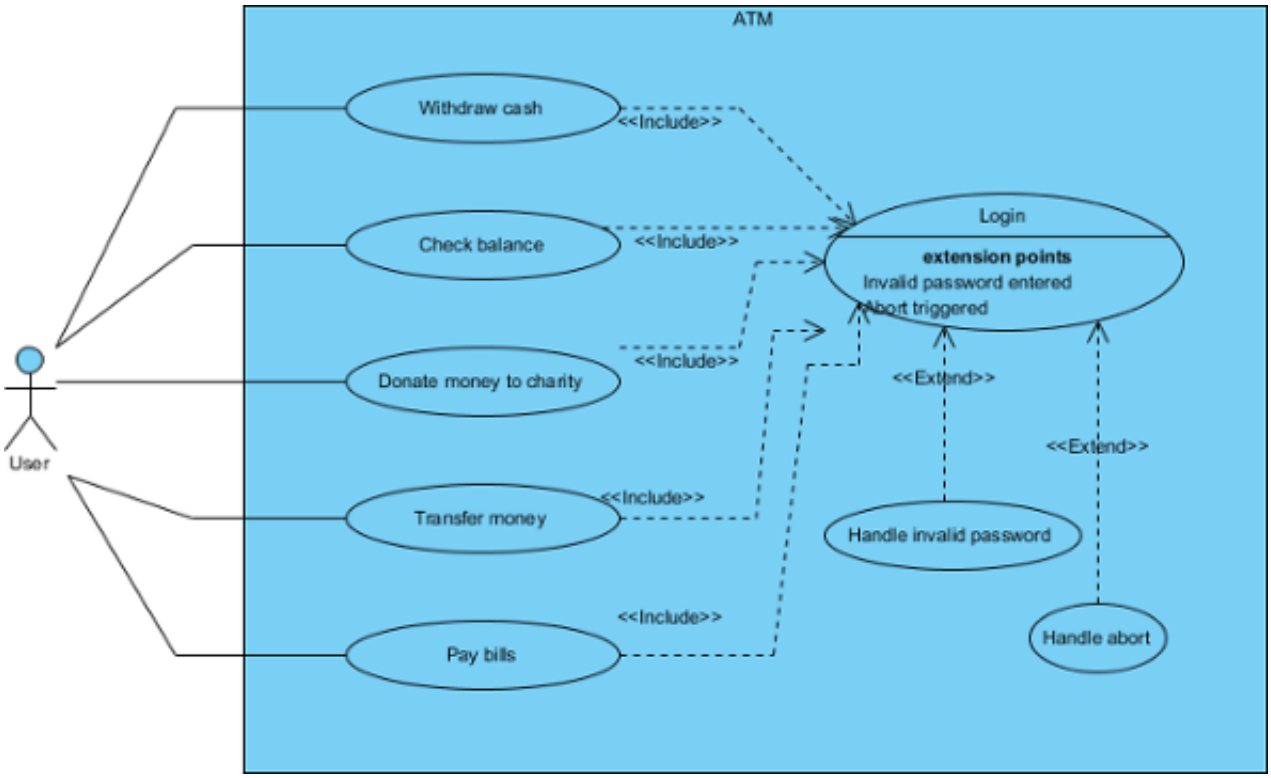
When you are working with a table, you can place the **${IMAGE}** field in a table cell to let Doc. Composer replace it with the diagram image of the querying diagram.

Note that **${IMAGE}** can only be used in a table cell.

Here is an example of **${IMAGE}**:

```
${IMAGE}
```

Here is the sample output:



Use Case Diagram

This is the syntax of an **${IMAGE}** field:

```
${IMAGE}
```

${PROPERTY}

When you are working with a table, you can place the **${PROPERTY}** field in a table cell to let Doc. Composer replace it with the property value of the querying diagram or element.

Note that **${PROPERTY}** can only be used in a table cell.

Here is an example of **${PROPERTY}**:

```
${PROPERTY}
```

Here is the sample output:

```
This is the description of use case.
```

This is the syntax of a **${ICON}** field:

```
${PROPERTY}
```

${TEXT}

The **${TEXT}** field is used when you need to include information that should be or can only be provided when generating document. A typical usage of **${TEXT}** is to request for project name.

In Doc. Composer, **${TEXT}** are represented as text fields. User can enter the value required by the **${TEXT}** field. When generating document, those **${TEXT}** will be replaced by the text entered.

Here is an example of **${TEXT}**:

```
${TEXT, "Project name"}
```

Here is the sample output:

```
Online Banking
```

This is the syntax of a **${TEXT}** field:

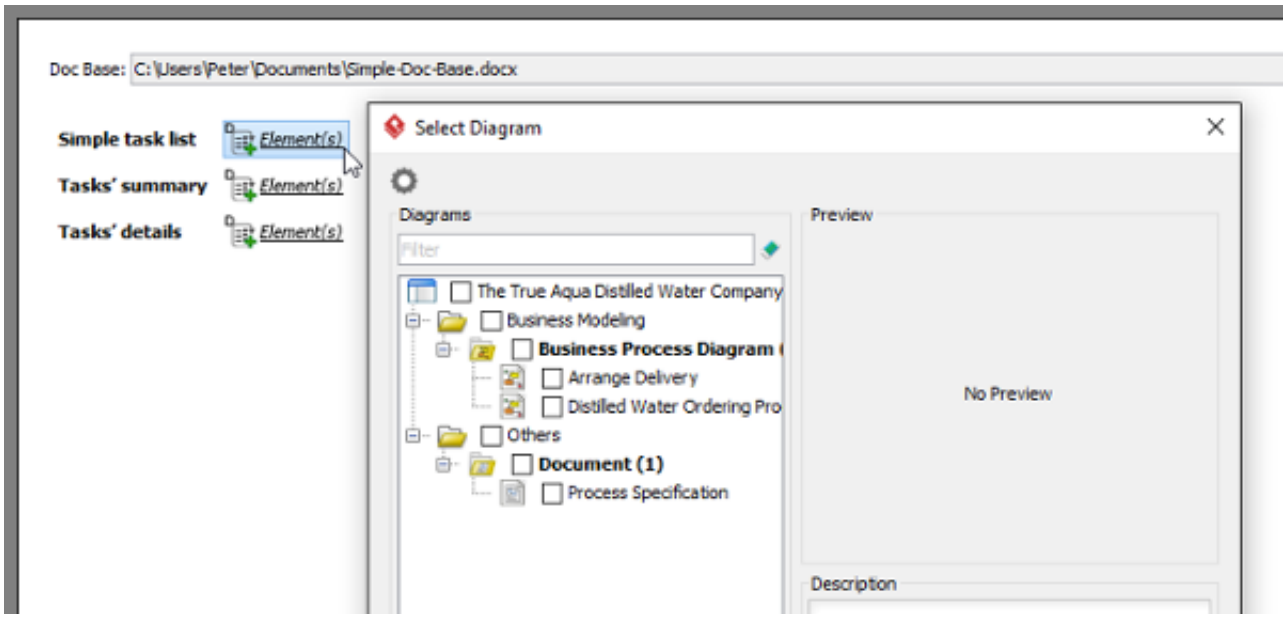
${TEXT, ***field_name*** }

This is a description of the various parts of a **${TEXT}** field:

- TEXT is to indicate that this is a **${TEXT}** field.
- ***field_name*** is a short description of the field (e.g. "Project name"). It can also be used as a reminder to provide certain kind of information (e.g. "Please enter the project name."). ***field_name*** must be unique within a Doc Base. If there are two or more fields having the same field name, content may be produced wrongly.

Reusability of Doc Fields

A Doc Base may contains many Doc Fields. If the Doc Fields use LoopInElement and LoopInDiagram, which require the selection of source in Doc. Composer, you (or the person who will generate document with the Doc Base) will then need to select model elements and diagrams again and again in Doc. Composer. This is not just time consuming but also error prone.



In order to solve this problem, you can reuse Doc Fields throughout a document. When you need to query data from a source that was expected by an earlier Doc Field, write the new Doc Field by using the same name as the previous one, like this:

${ELEMENT, "**Tasks in Main BPD**", "BPTask", LoopInDiagram, PROPERTY=name}

...

${ELEMENT, "**Tasks in Main BPD**", "BPSubProcess", LoopInDiagram, Basic}

..

${ELEMENT, "**Tasks in Main BPD**", "BPTask", LoopInDiagram, Details}

This three Doc Fields mean that I want to display the name of all tasks from a specific diagram (to be chosen in Doc. Composer). Then, I want to display the basic information of sub-processes on the same **diagram**. Finally, I want to display the details of tasks from again the **same diagram**.

By reusing Doc Fields, you just need to make the selection of source once. Subsequent Doc Fields will just apply the same selection. The key is to use same names for different Doc Fields.

Note that the same source must be supplied in order for the reusability to work. By "source", we are referring to the argument of a Doc Field that indicates the source from which to query elements/diagrams, such as One | Any | LoopInProject | LoopInElement | LoopInDiagram.