

How to: Arbeiten mit Sequence/Liste

EINFÜHRUNG UND ANWENDUNGSBEISPIELE



Vortragsziele

- Annahme: Arbeiten mit Listen == <#list>?

Vortragsziele

- Annahme: Arbeiten mit Listen == `<#list>`?
- Eine Alternative zur `<#list>`-Direktive aufzeigen
 - Übliche Funktionen für die Arbeit mit Listen vorstellen
 - Beispiele zur Weiterverwendung und Modifikation bieten
 - Für die Anwendung von Listen sensibilisieren
 - Präsentation und Synesty-Projekt als Nachschlagwerk

Quellen von Listen

WIE KOMME ICH AN EINE LISTE?

Manuell erstellte Listen

Syntax:

- Elemente komma-getrennt auflisten
- Auflistung mit eckigen Klammern umschließen

Beispiel – Auflistung mit... :

- ... Texten: ['a', 'b', 'c']
- ... Zahlen: [1, 2, 3, 4]
- ... Variablen: [Farbe1!, Farbe2!, Farbe3!]
- ... einer Mischung: ['a', 2, Farbe3!]

Manuell erstellte Listen - Ranges

Zahlenreihen kann man auch über Ranges erstellen

- Syntax:
 - Zwei Zahlenwerte **start** und **end** getrennt durch **..**
- Beispiele:
 - **1..3** => [1, 2, 3]
 - **3..1** => [3, 2, 1]
 - **start..end** => [1,2,3] (mit start = 1 und end = 3)

Variationen:

- ! für ein exklusives Ende: **1..!3** => [1,2]
- * für eine Liste fester Länge: **10..*3** => [10,11,12]

Manuell erstellte Listen - Ranges

Zahlenreihen kann man auch über Ranges erstellen

- Syntax:
 - Zwei Zahlenwerte **start** und **end** getrennt durch **..**
- Beispiele:
 - **1..3** => [1, 2, 3]
 - **3..1** => [3, 2, 1]
 - **start..end** => [1,2,3] (mit start = 1 und end = 3)

Anwendung:

- Slicing in Strings oder Listen: **"Fee Foo"[4..*3]** => Foo
- Wiederhole n-mal: **<#list (1..10) as i>Do 10 times</#list>**

Freemarker - Funktionen

Freemarker bietet einige Funktionen, die als Ergebnis eine Liste liefern:

- Split – Funktion:
 - trennt einen Text am vorgegebene Zeichen in eine Liste
 - Nützlich für Texte mit irgendeiner Form von Auslistungen
 - Beispiel: `'1,2,3'?split(',') => [1, 2, 3]`
- matches (Regex)
- word_list (== `split(" ")`)

Weitere Funktionen zum Verarbeiten von Listen:

- Meisten Built-Ins für Sequences / Listen
 - `filter`, `map`, Reverse, sort, sort_by, chunk, drop_while, ...
 - https://freemarker.apache.org/docs/ref_builtins_sequence.html

Synesty - Funktionen

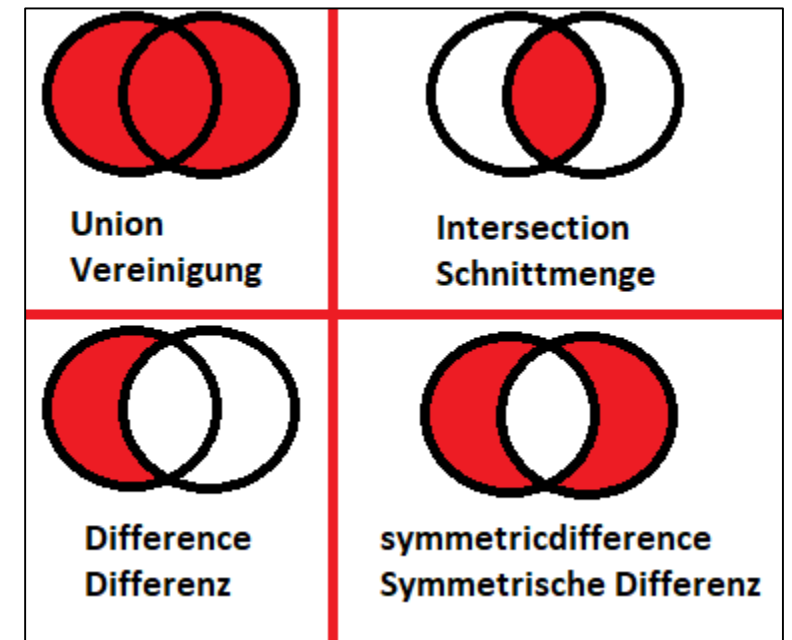
Spreadsheet-Datentypen sind in den meisten Fällen Listen.

Spreadsheet-Funktionen mit Listen als Ergebnis:

- `.getHeader()` – Liste alle Spaltentitel
- `.getRow()` – Liste aller Spreadsheet-Zeilen
- `.getCols()` – Liste aller Spalten in einer Spreadsheet-Zeile

Spreadsheet-Funktionen zum Verarbeiten von Listen:

- `deduplicate(LISTE)` – gibt eine Liste alle einzigartigen Elemente zurück
- Verschiedene Funktion aus der Mengenlehre:
 - union, intersection, difference, symmetricdifference
- Filter
 - limitierte Variante der Filter-Funktion von Freemarker



Anwendungsfälle

BEISPIEL #1
MEHRERE FARBSPALTEN ZUSAMMENFÜGEN

Mehrere Spalten zusammenfügen

Was ist die Ausgangssituation?

- Man hat n Eingangsspalten.
- Nur gefüllte Eingangsspalten sind interessant
- Diese sollen mit einem vorgegebenen Text verkettet werden.

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1	Grün	Blau	Pink	Grün und Blau und Pink
2	Blau	Schwarz		Blau und Schwarz
3	Rot			Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Einfacher IF-Ansatz funktioniert

```
1 <#if Farbe1! != "">${Farbe1!}</#if>  
2 <#if Farbe2! != ""> und ${Farbe2!}</#if>  
3 <#if Farbe3! != ""> und ${Farbe3!}</#if>
```

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1	Grün	Blau	Pink	Grün und Blau und Pink
2	Blau	Schwarz		Blau und Schwarz
3	Rot			Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Einfacher IF-Ansatz funktioniert... teilweise

```
1 <#if Farbe1! != "">${Farbe1!}</#if>  
2 <#if Farbe2! != ""> und ${Farbe2!}</#if>  
3 <#if Farbe3! != ""> und ${Farbe3!}</#if>
```

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	und Blau und Pink
2		Blau		und Blau
3			Rot	und Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Was ist die Situation?

- Man hat n Eingangsspalten.
- Nur gefüllte Eingangsspalten sind interessant
- Diese sollen mit einem vorgegebenen Text verkettet werden.

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	Blau und Pink
2		Blau		Blau
3			Rot	Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Was ist die Situation?

- Man hat n Eingangsspalten. => [Farbe1!, Farbe2!, Farbe3!]
- Nur gefüllte Eingangsspalten sind interessant
- Diese sollen mit einem vorgegebenen Text verkettet werden.

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	Blau und Pink
2		Blau		Blau
3			Rot	Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Was ist die Situation?

- Man hat n Eingangsspalten. => [Farbe1!, Farbe2!, Farbe3!]
- Nur gefüllte Eingangsspalten sind interessant => ???
- Diese sollen mit einem vorgegebenen Text verkettet werden.

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	Blau und Pink
2		Blau		Blau
3			Rot	Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Freemarker-Funktion - Filter

Die Filter-Funktion

- wird auf eine Liste angewandt.
- evaluiert für jedes Element in der Liste eine Funktion mit Boolean-Ergebnis
- erstellt eine neue Liste mit den Elementen, für die die evaluierte Funktion true zurück gibt

Syntax: [...]?filter(**Variable** -> **Ausdruck mit Boolean-Ergebnis**) aka. Lambda-Ausdruck

Variable:

- definiert den Namen, unter dem das Element adressiert werden kann

Ausdruck mit Boolean-Ergebnis:

- definiert die zu evaluierende Funktion (meist im Bezug zur Variable)

Bsp 1: Liste von nicht-leeren Texten: [...]?filter(x -> x != "")

Bsp 2: Liste von Texten, die "farbe" enthalten: [...]?filter(element -> element?contains("farbe"))

Mehrere Spalten zusammenfügen

Was ist die Situation?

- Man hat n Eingangsspalten. => [Farbe1!, Farbe2!, Farbe3!]
- Nur gefüllte Eingangsspalten sind interessant => ?filter(x -> x?trim != "")
- Diese sollen mit einem vorgegebenen Text verkettet werden.

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	Blau und Pink
2		Blau		Blau
3			Rot	Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Mehrere Spalten zusammenfügen

Was ist die Situation?

- Man hat n Eingangsspalten. => [Farbe1!, Farbe2!, Farbe3!]
- Nur gefüllte Eingangsspalten sind interessant => ?filter(x -> x?trim != "")
- Diese sollen mit einem vorgegebenen Text verkettet werden. => ?join(" und ")

[Farbe1!, Farbe2!, Farbe3!]?filter(x -> x?trim != "")?join(" und ")

ID	Farbe1	Farbe2	Farbe3	Ergebnis
1		Blau	Pink	Blau und Pink
2		Blau		Blau
3			Rot	Rot
4	Schwarz	Gelb	Violet	Schwarz und Gelb und Violet
5	Rot	Grün		Rot und Grün
6	Gelb	Blau	Rot	Gelb und Blau und Rot
7	Grün	Weiß		Grün und Weiß
8	Blau	Schwarz		Blau und Schwarz
9	Weiß			Weiß
10	Schwarz	Gelb		Schwarz und Gelb

Anwendungsfälle

BEISPIEL #2
KOMMA-GETRENNTE EIGENSCHAFTEN

Komma-Getrennte Eigenschaften

Was ist die Ausgangssituation?

- Langer Text mit vielen unterschiedlichen Eigenschaften

Sonstige Eigenschaft=ABC, Farbe1=Grün, Größe=30cm, Sonstige Eigenschaft=ABC, Sonstige Eigenschaft=ABC, Farbe2=Rot, Sonstige Eigenschaft=ABC, Farbe3=Blau

Komma-Getrennte Eigenschaften

Was ist die Ausgangssituation?

- Langer Text mit vielen unterschiedlichen Eigenschaften

Sonstige Eigenschaft=ABC, Farbe1=Grün, Größe=30cm, Sonstige Eigenschaft=ABC, Sonstige Eigenschaft=ABC, Farbe2=Rot, Sonstige Eigenschaft=ABC, Farbe3=Blau, Sonstige Eigenschaft=ABC

- Auch hier sollen die Farben ausgelesen werden.

Komma-Getrennte Eigenschaften

Was ist die Ausgangssituation?

- Langer Text mit vielen unterschiedlichen Eigenschaften

Sonstige Eigenschaft=ABC,Farbe1=Grün,Größe=30cm,Sonstige Eigenschaft=ABC,Sonstige Eigenschaft=ABC,Farbe2=Rot,Sonstige Eigenschaft=ABC,Farbe3=Blau,Sonstige Eigenschaft=ABC

- Auch hier sollen die Farben ausgelesen werden.

```
2 <#list eigenschaften_text?split(",") as eigenschaft>
3   <#if eigenschaft?starts_with("Farbe")>
4     ${eigenschaft}<#sep>,
5   </#if>
6 </#list>
```

1 Farbe1=Grün,
2 Farbe2=Rot,
3 Farbe3=Blau,

The diagram illustrates the transformation of a long, comma-separated string of properties into a list of color properties. An orange arrow points from the code block on the left to the resulting list on the right. The list contains three items: 'Farbe1=Grün,', 'Farbe2=Rot,', and 'Farbe3=Blau,'.

Komma-Getrennte Eigenschaften

Was ist die Ausgangssituation?

- Langer Text mit vielen unterschiedlichen Eigenschaften

Sonstige Eigenschaft=ABC,Farbe1=Grün,Größe=30cm,Sonstige Eigenschaft=ABC,Sonstige Eigenschaft=ABC,Farbe2=Rot,Sonstige Eigenschaft=ABC,Farbe3=Blau,Sonstige Eigenschaft=ABC

- Auch hier sollen die Farben ausgelesen werden.
- Filter statt <#if>-Klausel

```
2 <#list eigenschaften_text?split(",")?filter(prop -> prop?starts_with("Farbe")) as eigenschaft>  
3   ${eigenschaft}<#sep>,  
4 </#list>
```

```
1 Farbe1=Grün,  
2 Farbe2=Rot,  
3 Farbe3=Blau
```


Komma-Getrennte Eigenschaften


Was ist die Ausgangssituation?

- Langer Text mit vielen unterschiedlichen Eigenschaften

Sonstige Eigenschaft=ABC,Farbe1=Grün,Größe=30cm,Sonstige Eigenschaft=ABC,Sonstige Eigenschaft=ABC,Farbe2=Rot,Sonstige Eigenschaft=ABC,Farbe3=Blau,Sonstige Eigenschaft=ABC

- Auch hier sollen die Farben ausgelesen werden.
- Filter statt <#if>-Klausel
- Join statt <#list>

```
2  ${eigenschaften_text?split(",")?filter(prop -> prop?starts_with("Farbe"))?join(",  
3  ")}
```



```
1  Farbe1=Grün,  
2  Farbe2=Rot,  
3  Farbe3=Blau
```

Komma-Getrennte Eigenschaften

```
2 <#list eigenschaften_text?split(",")?filter(prop -> prop?starts_with("Farbe")) as eigenschaft>  
3 `${eigenschaft?keep_after("=")}`<#sep>,  
4 </#list>
```



1	Grün,
2	Rot,
3	Blau

Wie wendet man eine Funktion ohne <#list> an? => ???

Freemarker-Funktion - Map

Die Map-Funktion

- wird auf eine Liste angewandt.
- evaluiert für jedes Element in der Liste eine Funktion mit beliebigem Ergebnis
- erstellt eine neue Liste mit den Ergebnis der Funktion

Syntax: `?map(Variable -> Ausdruck mit beliebigen Ergebnis)` aka. Lambda-Ausdruck

Bsp 1: Text kürzen: `[...]?map(x -> x?keep_after("="))`

Bsp 2: Text ergänzen: `[...]?map(element -> "Textbaustein:" + element)`

Bsp 3: Text kürzen: `[...]?map(x -> x[0..*10])`

Bsp 4: Key-Value-Liste aller Spalten: `row.getCols()?map(col -> col.title + "=" + col.get())`

Komma-Getrennte Eigenschaften

```
2 <#list eigenschaften_text?split(",")?filter(prop -> prop?starts_with("Farbe")) as eigenschaft>  
3   ${eigenschaft?keep_after("=")}<#sep>,  
4 </#list>
```

1	Grün,
2	Rot,
3	Blau

Wie wendet man eine Funktion ohne <#list> an? => map

```
2   ${eigenschaften_text?split(",")?filter(prop -> prop?starts_with("Farbe"))?map(prop -> prop?keep_after("="))?join(",  
3   ")}  
3   "}
```

Anwendungsfälle

BEISPIEL #3
SPREADSHEET-SPALTEN ZUSAMMENFASSEN

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3			Rot	80	Gramm	quadratisch		
4	Schwarz	Gelb	Violet	119		oval	Stabil	
5	Rot	Grün		22	Kilogramm	oval		
6	Gelb	Blau	Rot	0,001				
7	Grün	Weiß		465	Gramm			
8	Blau	Schwarz		3375	Gramm	rund		Mauris et vestibulum nisl.
9	Weiß		Schwarz	1,157	Kilogramm			Maecenas scelerisque libero sit amet elementum lobortis.
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
...								
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte aller Spalten sollen zusammengefasst:

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
							...	
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte aller Spalten sollen zusammengefasst:

Skript:

```
${row.getCols()?.map(col -> col.title + "=" + col.get())?.join("|")}
```

```
=>ID=1 | Farbe1= | Farbe2=Blau | Farbe3=Pink | Gewicht=1,2 | Gewichteinheit=Kilogramm | Form=rund | Qualität  
smerkmal=Haltbar | Langtext=
```


Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
							...	
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte von Spalten sollen zusammengefasst:

- für gefüllte Spalten `=> ?filter(col -> col.get() != "")`

Skript:

```
`${row.getCols().filter(col -> col.get() != "").map(col -> col.title + "=" + col.get()).join("|")}`
```

```
=>ID=1 | Farbe2=Blau | Farbe3=Pink | Gewicht=1,2 | Gewichteinheit=Kilogramm | Form=rund | Qualitätsmerkmal=Haltbar
```

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
...								
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte von Spalten sollen zusammengefasst:

- für gefüllte Spalten `=> ?filter(col -> col.get() != "")`
- für Farbspalten `=> ?filter(col -> col.title?starts_with("Farbe"))`

Skript:

```
row.getCols().map(col -> col.title + "=" + col.get())?join("|")  
=> Farbe1=|Farbe2=Blau|Farbe3=Pink
```

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
							...	
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte von Spalten sollen zusammengefasst:

- für gefüllte Spalten `=> ?filter(col -> col.get() != "")`
- für Farbspalten `=> ?filter(col -> col.title?starts_with("Farbe"))`
- für gefüllte Farbspalten `=> ?filter(col -> col.get() != "" && col.title?starts_with("Farbe"))`

Skript:

```
#{row.getCols()?map(col -> col.title + "=" + col.get())?join("|")}
```

```
=> Farbe2=Blau|Farbe3=Pink
```

Spreadsheet-Spalten zusammenfassen

ID	Farbe1	Farbe2	Farbe3	Gewicht	Gewichteinheit	Form	Qualitätsmerkmal	Langtext
1		Blau	Pink	1,2	Kilogramm	rund	Haltbar	
2		Blau		13	Kilogramm	rechteckig		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
							...	
10	Schwarz	Gelb		30	Gramm	quadratisch		Aliquam in erat vel neque ultricies hendrerit facilisis et dolor.

Werte von Spalten sollen zusammengefasst:

- für gefüllte Spalten `=> ?filter(col -> col.get() != "")`
- für Farbspalten `=> ?filter(col -> col.title?starts_with("Farbe"))`
- für gefüllte Farbspalten `=> ?filter(col -> col.get() != "" && col.title?starts_with("Farbe"))`
- für gefüllten Spalten außer ID `=> ?filter(col -> col.get() != "" && col.title != "ID")`

Skript:

```
#{row.getCols().map(col -> col.title + "=" + col.get())?join("|")}
```

```
=>Farbe2=Blau|Farbe3=Pink|Gewicht=1,2|Gewichteinheit=Kilogramm|Form=rund|Qualitätsmerkmal=Haltbar
```

Anwendungsfälle

GRUNDOPERATIONEN UND KLEINE BEISPIEL



Grundoperationen

Listen vereinen:	$[1,2,3] + [4,5,6] \Rightarrow [1,2,3,4,5,6]$
Listen sortieren:	$[5,3,6,1,2,4]?sort \Rightarrow [1,2,3,4,5,6]$
Ist ein bestimmtes Element enthalten:	$[1,2,3,4,5,6]?seq_contains(5) \Rightarrow true$
Größe einer Liste bestimmen:	$[1,2,3,4,5,6]?size \Rightarrow 6$
Größte Element bestimmen:	$[1,2,3,4,5,6]?max \Rightarrow 6$
Kleinste Element bestimmen:	$[1,2,3,4,5,6]?min \Rightarrow 1$

Verschiedene kleine Beispiel

Niedrigsten Wert aus mehreren Spalten auswählen:

```
$$([[LIFE_TIME_TYPE_LED!, LIFE_TIME_MODULES!, LIFE_TIME_STRIPS!,LIFE_TIME_BULB!]  
?filter(x -> x?trim != "")?map(x -> x?number))?min}
```

Anzahl an Übermaße bestimmen

```
$$([PACKAGE_LENGTH!,PACKAGE_WIDTH!,PACKAGE_HEIGHT!]?filter(x -> x?trim != "")  
?map(x -> x?trim?number)?filter(x -> x gt 63.5)?size}
```

Erster gefüllter Wert

```
[...]?filter(x -> x?trim != "")?first}   oder [...]?filter(x -> x?trim != "")[0]}
```

Verschiedene kleine Beispiel #2

Liste aller leeren Spalten:

```
${row.getCols()?.filter(col -> col.get() == "")?.map(col -> col.getTitle())?.join(",")}
```

Prüfen, ob zwei Listen dieselben Elemente enthalten

```
${symmetricdifference(listeA, listeB)?.size == 0}
```


Verschiedene kleine Beispiel #3

XML-Produktion:

```
27     <#list family_row.getCols()?filter(col -> col.title!?.starts_with("Pictogram_") && col.get()?.trim != "") as pictogram_col>
28         <Icon id="{pictogram_col.title?.keep_after_last("_")}">
29             <Path>{pictogram_col.get()}</Path>
30         </Icon>
31     </#list>
```