

LimeSurvey-Tutorial

Mehrfachnennung
Mehrfache kurze Texte
Slider
Drop-Downs

September 2021 – November 2023

Inhaltsverzeichnis

Vorbemerkung.....	4
1 Mehrfachnennung.....	5
1.1 Header in Mehrfachnennung.....	5
1.1.1 Anwendungsbeispiel.....	5
1.1.2 Implementierung.....	6
1.1.3 Nebenbemerkung.....	8
1.2 Mehrfachnennung mit mehreren Levels.....	9
1.2.1 Anwendungsbeispiel.....	9
1.2.2 Implementierung.....	10
1.3 Fixierte Antworten am Ende einer Mehrfachnennung.....	14
1.3.1 Anwendungsbeispiel.....	14
1.3.2 Implementierung.....	15
1.4 Beschränkung der anwählbaren Optionen.....	17
1.4.1 Anwendungsbeispiel.....	17
1.4.2 Implementierung.....	18
1.5 Fixe umfrageweite Randomisierung.....	19
1.5.1 Anwendungsbeispiel.....	19
1.5.2 Implementierung.....	20
2 Mehrfachnennung mit Kommentar.....	24
2.1 Ausgeblendete Kommentarfelder.....	24
2.1.1 Anwendungsbeispiel.....	24
2.1.2 Implementierung.....	25
2.2 Geänderte Kommentarfeldlänge, Header.....	26
2.2.1 Anwendungsbeispiel.....	26
2.2.2 Implementierung.....	27
2.3 Mehrfachnennung mit Kommentar, mehrere Spalten.....	28
2.3.1 Anwendungsbeispiel.....	28
2.3.2 Implementierung.....	28
2.4 Textausrichtung.....	30
2.4.1 Anwendungsbeispiel.....	30
2.4.2 Implementierung.....	31
2.5 Mehrfachnennung, mehrspaltig mit Header.....	32
2.5.1 Anwendungsbeispiel.....	32
2.5.2 Implementierung.....	33
3 Mehrfache kurze Texteingabe.....	36
3.1 Mit Drop-Down und Datepicker.....	36
3.1.1 Anwendungsbeispiel.....	36
3.1.2 Implementierung.....	37
3.2 autocomplete.....	40
3.2.1 Anwendungsbeispiel.....	40
3.2.2 Implementierung.....	41
3.2.2.1. als Matrix.....	41
3.2.2.2. als Textdatei.....	42
3.2.3 Was könnte dagegen sprechen.....	43
3.3 Verschiedene Suffixe der Teilfragen.....	44

3.3.1 Anwendungsbeispiel.....	44
3.3.2 Implementierung.....	44
3.4 „Dynamische“ Anzeige der Eingabefelder.....	45
3.4.1 Anwendungsbeispiel.....	45
3.4.2 Implementierung.....	46
4 Slider.....	50
4.1 Slider mit Ticks und Bildern/Smileys.....	50
4.1.1 Anwendungsbeispiele.....	50
4.1.2 Implementierung.....	51
4.2 Slider mit wechselnden Tooltips.....	57
4.2.1 Anwendungsbeispiel.....	57
4.2.2 Implementierung.....	58
4.3 Slider mit mehreren Bereichen.....	59
4.3.1 Anwendungsbeispiel.....	59
4.3.2 Implementierung.....	59
5 Drop-Down.....	63
5.1 Drop-Down als Option in Einfach- und Mehrfachnennung.....	63
5.1.1 Anwendungsbeispiel.....	63
5.1.2 Implementierung.....	64
5.1.2.1. Einfachnennung.....	64
5.1.2.2. Mehrfachnennungen.....	66
6 Anhang.....	68
6.1 Iss-Export einer Beispiel-Datei.....	68
6.2 Zusätzliche benötigte Bibliotheken und Dateien.....	68
6.2.1 Beispiel-Text-Dateien für „autocomplete“.....	68
6.2.2 Zusätzliche Bibliotheken für „autocomplete“.....	68

Vorbemerkung

Diese kleinen Tutorials sollen Möglichkeiten aufzeigen, durch Einsatz von javascript und css Effekte bei der Darstellung von Fragen zu zeigen, die den Teilnehmern die Beantwortung erleichtert.

Obwohl einige der Beispiele auch mit plugins bzw. zu installierenden Frage-Designs realisierbar wären, habe ich bewusst darauf verzichtet.

Viele der studentischen Nutzer haben weder Rechte, plugins zu installieren noch am Theme Änderungen vorzunehmen.

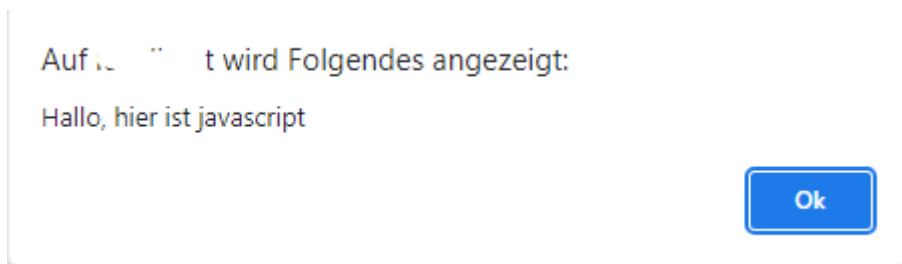
Daher sind alle Beispiele alleine durch Einfügen von javascript bzw. css im Fragetext erstellt, von denen die meisten („Ehre wem Ehre gebührt“) ursprünglich von Tony Partner (tpartner) stammem.

Da nahezu alle gezeigten Beispiele mit javascript realisiert werden, ist natürlich die Grundvoraussetzung, dass dieses eingesetzt werden kann.

Ein einfacher Test ist dies in den Quellcode der ersten Frage einzufügen

```
<script type="text/javascript" charset="utf-8">
$(document).on('ready ajax:scriptcomplete',function(){
    alert("Hallo, hier ist javascript");
});
</script>
```

Wenn dann dieses Fenster aufploppt, ist alle in Ordnung; wenn nicht, ...



Die dargestellten Codes sollten nicht aus diesem Text in LimeSurvey kopiert werden.

Es könnten sich noch Reste der Formatierung im Text befinden, der den Code dann unbrauchbar macht.

Daher bitte diese Codestücke immer aus der Beispielstudie übernehmen.

Bitte beachten:

Falls eine php Version 8.0 und höher installiert ist, funktionieren einige Beispiele nicht in der Version 3.x. von LimeSurvey (z.B. autocomplete)

Und dann noch dieses:

Solutions, code and workarounds presented in these text are given without any warranty, implied or otherwise.

Lösungen, Code und „Work-arounds“, die in diesem Text präsentiert werden, werden ohne jegliche stillschweigende oder sonstige Gewährleistung gegeben.

1 Mehrfachnennung

1.1 Header in Mehrfachnennung

1.1.1 Anwendungsbeispiel

In Mehrfachnennungsfragen kann es manchmal hilfreich sein, die Antwortoptionen nach Teilbereichen zu untergliedern, so dass die Übersichtlichkeit für den Teilnehmer verbessert wird.

Header in Mehrfachnennung

Fleisch
☐ Ameisensteak
☐ Biberfilet
☐ Chinchillalende

Gemüse
☐ Artischocken
☐ Bohnen
☐ Chicoree

Obst
Das ist eine Unterzeile
☐ Ananas
☐ Brombeeren

Gegenüber der Lösung in Kapitel 2.b. ist diese dann vorteilhafter, wenn Teilfragen durch vorherige Angaben gar nicht angezeigt werden. Sollten dadurch alle Obstsorten wegfallen, so wird der Header mit derselben Teilfragenrelevanzgleichung ausgeblendet.

1.1.2 Implementierung

Zunächst einmal: Diese „Header“ sind ganz normale Teilfragen. Mit dem javascript code wird lediglich das Kästchen zum Anwählen entfernt (und dann wird noch ein bisschen „Schönheit“ gemacht).

Code	Teilfrage
H1	Fleisch
SQ001	Ameisensteak
SQ002	Biberfilet
SQ003	Chinchillalende

Wie man hier sieht, habe ich für das Styling der „Header“ css-Klassen „myHeader“ und „mySubHeader“ benutzt, die dann später im css-Teil definiert werden

Außerdem habe ich die Codes verschieden von den Codes der „normalen“ Teilfragen gewählt; dies kann von Vorteil sein, wenn man mithilfe von „self“ und „that“ später einmal auf spezielle Teile zugreifen will.

Der javascript Code (der wie immer in den Fragetext – im Quellcode-Modus – eingetragen wird) ist sehr übersichtlich.

Wichtig ist die Stelle : **eq(x)**

Hier wird definiert, welche Teilfragen als Header angezeigt werden sollen (hier die 1., die 5. und die 9.)

Nicht verwirren lassen; die Zählung hier beginnt bei 0.

```
<script charset="utf-8" type="text/javascript">
  $(document).ready(function() {
    $('#question{QID} .question-item:eq(0)').addClass('hide-pseudo-
elements').find('input.checkbox').remove();
    $('#question{QID} .question-item:eq(4)').addClass('hide-pseudo-
elements').find('input.checkbox').remove();
    $('#question{QID} .question-item:eq(8)').addClass('hide-pseudo-
elements').find('input.checkbox').remove();
  });
</script>
```

Grundsätzlich genügt dieses bisschen css.

Oben im javascript wurde den Elementen ja die Klasse „hide-pseudo-elements“ hinzugefügt.

Auf diese wird jetzt zurückgegriffen, um die Kästchen zu entfernen und den Text weiter nach links zu rücken. Diese Zeile „margin-left: -40px;“ muss möglicherweise je nach Theme angepasst werden.

```
<style type="text/css">
.hide-pseudo-elements label::before,
.hide-pseudo-elements label::after
{
    display: none;
}
.hide-pseudo-elements .label-text
{
    margin-left: -40px;
}
</style>
```

Und nun folgt noch das Styling der Texte mit den Klassen „myHeader“ und „mySubHeader“. Dies ist natürlich optional. Man könnte dies auch in den Text der Teilfrage hineinschreiben und natürlich hier nach Gusto Farbe, Größe, Hintergrund, u.ä. anpassen.

```
<style type="text/css">
.myHeader {
    color: maroon;
    font-size: 130%;
    font-weight: bold;
    background-color: #eeeeee;
    border: 1px solid #cccccc;
    padding: 5px;
    margin-left: -35px;
}
.mySubHeader {
    color: green;
    font-size: 60%;
    font-weight: normal;
    font-style: italic;
}
</style>
```

1.1.3 Nebenbemerkung

Oft wird moniert, dass zwischen den Items in Mehrfachnennungen (aber auch Einfachnennungen) sehr viel Platz ist. Dies kann man aber ändern.

Mit diesem css-Code, den man am besten ans Ende der „custom.css“ platziert (oder je nach Bedarf auch in den Fragetext, wird der Abstand angepasst.

```
<style>
  li.radio-item, li.checkbox-item, li.radio-text-item, li.checkbox-text-item {
    margin-bottom: 0em;
  }
</style>
```

Wie man sieht, werden hiermit sowohl „Radio-Buttons“ wie auch „Checkboxes“ angesprochen.

Hier ein Vergleich (ja, man kann sogar negative Werte angeben)

Standard

Header in Mehrfachnennung

Fleisch
☐ Ameisensteak
☐ Biberfilet
☐ Chinchillalende

Gemüse
☐ Artischocken
☐ Bohnen
☐ Chicoree

Obst
Das ist eine Unterzeile
☐ Ananas
☐ Brombeeren

0em

Header in Mehrfachnennung

Fleisch
☐ Ameisensteak
☐ Biberfilet
☐ Chinchillalende

Gemüse
☐ Artischocken
☐ Bohnen
☐ Chicoree

Obst
Das ist eine Unterzeile
☐ Ananas
☐ Brombeeren

-1.0em

Header in Mehrfachnennung

Fleisch
☐ Ameisensteak
☐ Biberfilet
☐ Chinchillalende

Gemüse
☐ Artischocken
☐ Bohnen
☐ Chicoree

Obst
Das ist eine Unterzeile
☐ Ananas
☐ Brombeeren

Das letzte Beispiel ist wohl etwas übertrieben; es zeigt aber den Effekt. Dies liegt nun in der Hand des Nutzers.

1.2 Mehrfachnennung mit mehreren Levels

1.2.1 Anwendungsbeispiel

Einen ähnlichen Sinn und Zweck wie die oben gezeigten Header hat auch diese Anzeige einer Mehrfachnennung mit mehreren Levels; sie soll die Übersichtlichkeit fördern, indem der Teilnehmer zunächst nur die Oberbegriffe sieht, und sich dann zu den Unterbegriffen „durchklicken“ kann.

Man beginnt also ganz harmlos, und bei Klick werden die entsprechenden weiteren Ebenen aufgeklappt.

Mehrfachnennung drei Ebenen	Mehrfachnennung drei Ebenen
<input type="checkbox"/> Ameise ...	<input checked="" type="checkbox"/> Ameise ...
<input type="checkbox"/> Biber	<input checked="" type="checkbox"/> Ameisenbär
<input type="checkbox"/> Dachs	<input type="checkbox"/> Roter Ameisenbär
<input type="checkbox"/> Andere	<input type="checkbox"/> Blauer Ameisenbär
	<input type="checkbox"/> Gelber Ameisenbär
	<input type="checkbox"/> Ameisenlöwe
	<input type="checkbox"/> Biber
	<input checked="" type="checkbox"/> Dachs
	<input type="checkbox"/> Weißdachs
	<input type="checkbox"/> Frechdachs
	<input type="checkbox"/> Schwarzdachs
	<input checked="" type="checkbox"/> Andere
	<input checked="" type="checkbox"/> Elefant
	<input type="checkbox"/> Langrüssler
	<input type="checkbox"/> Kurzrüssler
	<input type="checkbox"/> Giraffe

1.2.2 Implementierung

Die Implementierung im javascript Code, der in den Fragetext (im Quellcode-Modus) einzutragen ist, beruht auf der Codierung der Teilfragen.

Zum Beispiel hier:

SQ1	Ameise ...
SQ11	Ameisenbär
SQ111	Roter Ameisenbär
SQ112	Blauer Ameisenbär
SQ113	Gelber Ameisenbär
SQ12	Ameisenlöwe
SQ2	Biber
SQ21	Graubiber
SQ22	Schwarzbiber
SQ3	Dachs
SQ31	Weißdachs
SQ32	Frechdachs
SQ321	wenig frech
SQ322	mittel frech
SQ323	sehr frech
SQ33	Schwarzdachs
SQ4	Andere
SQ41	Elefant
SQ411	Langrüssler
SQ412	KurZRüssler
SQ42	Giraffe
SQ43	Habicht

Man sieht deutlich die hierarchische Struktur.

Diese wird nun im javascript Code hier verwendet:

```
// First-level sub-question codes
['SQ1', 'SQ2', 'SQ3', 'SQ4'],

// Second-level sub-question codes
['SQ11', 'SQ12', 'SQ21', 'SQ22', 'SQ31', 'SQ32', 'SQ33', 'SQ41', 'SQ42',
'SQ43'],

// Third-level sub-question codes
['SQ111', 'SQ112', 'SQ113', 'SQ321', 'SQ322', 'SQ323', 'SQ411', 'SQ412']
```

```

<script type="text/javascript" charset="utf-8">

// A function to handle "child" checkboxes
function dependantCheckboxes(qID, primaryCodes, secondaryCodes, tertiaryCodes) {
    // Identify the elements and assign classes/attributes
    var thisQuestion = $('#question'+qID);
    thisQuestion.addClass('with-dependants');
    $.each(primaryCodes, function(i, val) {
        var thisItem = $('li[id$="X'+qID+val+']');
        $(thisItem).addClass('level-1 parent-item').attr('data-code', val).attr('data-level', '1');
    });
    $.each(secondaryCodes, function(i, val) {
        var thisItem = $('li[id$="X'+qID+val+']');
        var thisParent1 = $(thisItem).prevAll('li[data-level="1"]:eq(0)');
        $(thisItem).addClass('level-2 parent-item child-item').attr('data-code', val).attr('data-level', '2').attr('data-parent-1', $(thisParent1).attr('data-code'));
    });
    $.each(tertiaryCodes, function(i, val) {
        var thisItem = $('li[id$="X'+qID+val+']');
        var thisParent1 = $(thisItem).prevAll('li[data-level="1"]:eq(0)');
        var thisParent2 = $(thisItem).prevAll('li[data-level="2"]:eq(0)');
        $(thisItem).addClass('level-3 child-item').attr('data-code', val).attr('data-level', '3').attr('data-parent-1', $(thisParent1).attr('data-code')).attr('data-parent-2', $(thisParent2).attr('data-code'));
    });

    // A function to handle the states of child items
    function handleChildren(el) {
        var thisitem = $(el).closest('li');
        var thisCode = $(thisitem).attr('data-code');
        var thisLevel = $(thisitem).attr('data-level');
        var thisChildren = $('li[data-level="'+(Number(thisLevel)+1)+'"] [data-parent-'+thisLevel+'="'+thisCode+'"]', thisQuestion);

        // Hide/show the secondary answers accordingly
        if (!$el.is(':checked')) {
            $(thisChildren).fadeOut(300, function(e) {
                $('input:checkbox', thisChildren).prop('checked', false).trigger('change');
            });
        } else {
            $(thisChildren).fadeIn(300);
        }
    }

    // Initial states of the secondary answers
    $('.parent-item input:checkbox', thisQuestion).each(function(i) {
        handleChildren($(this));
    });

    // A listener on the primary answer to show or hide secondary answers
    $('.parent-item input:checkbox', thisQuestion).on('change', function(e) {
        handleChildren($(this));
    });
}

```

```

});
}

$(document).on('ready ajax:scriptcomplete',function(){
    dependantCheckboxes(
        {QID},
        // First-level sub-question codes
        ['SQ1', 'SQ2', 'SQ3', 'SQ4'],

        // Second-level sub-question codes
        ['SQ11', 'SQ12', 'SQ21', 'SQ22', 'SQ31', 'SQ32', 'SQ33', 'SQ41', 'SQ42', 'SQ43'],

        // Third-level sub-question codes
        ['SQ111', 'SQ112', 'SQ113', 'SQ321', 'SQ322', 'SQ323', 'SQ411', 'SQ412']
    );
});
</script>

```

Natürlich gibt es auch noch ein bisschen css, um die Einrückung zu steuern. Die zweite Ebene wird also um 2,5em eingerückt, die dritte um 5,0em. Kann natürlich geändert werden.

```

<style type="text/css">.with-dependants .child-item {
    display: none;
}

.with-dependants li.level-2 {
    margin-left: 2.5em;
}

.with-dependants li.level-3 {
    margin-left: 5em;
}
</style>

```

Falls man nur zwei Ebenen anzeigen möchte, so genügt es, die Zeile mit den Codes der dritten Ebene leer zu lassen. Zum Beispiel so:

```

// First-level sub-question codes
['SQ1', 'SQ2', 'SQ3', 'SQ4'],

// Second-level sub-question codes
['SQ11', 'SQ12', 'SQ21', 'SQ22', 'SQ31', 'SQ32', 'SQ33', 'SQ41', 'SQ42',
'SQ43'],

// Third-level sub-question codes
[]

```

In der Beispielstudie befindet sich noch ein erweitertes Beispiel einer solchen Implementation.

Und hier schon einmal ein kleines Video.

[Klick für Beispiel-Video](#)

1.3 Fixierte Antworten am Ende einer Mehrfachnennung

1.3.1 Anwendungsbeispiel

Wenn man in Mehrfachnennungen (aber auch Einfachnennungen) die zusätzliche Option „Sonstige“ einschaltet, und auch – wie es ja sein sollte – die Antwortmöglichkeit „Keine davon“ anbietet, so sieht die Frage in etwa so aus.

Feste Antworten am Ende einer Mehrfachnennung mit "Sonstigen"

☐ Adalbert Ameise

☐ Berta Biber

☐ Carlo Chinchilla

☐ Doris Dachs

☐ Keiner davon

☐ Anderer Spieler

Es ist wohl nicht sehr schön, dass die Option „Keiner davon“ vor der Option „Anderer Spieler“ angezeigt wird.

Anmerkung: In Version 5.x. gibt es eine direkte Einstellungsmöglichkeit, um die Platzierung von „Sonstigen“ zu bestimmen.

Und wenn man die Reihenfolge randomisiert, wird es „noch verrückter“

Feste Antworten am Ende einer Mehrfachnennung mit "Sonstigen"

☐ Doris Dachs

☐ Keiner davon

☐ Carlo Chinchilla

☐ Berta Biber

☐ Adalbert Ameise

☐ Anderer Spieler

Dies liegt natürlich daran, dass „Keiner davon“ eine ganz normale Teilfrage ist, die also auch normal

mit rotiert wird.

Es wäre also schön man könnte

- „Keiner davon“ immer als letzte Antwortoption haben
- „Anderer Spieler“ immer als vorletzte

Feste Antworten am Ende einer Mehrfachnennung mit "Sonstigen"

☐ Doris Dachs

☐ Berta Biber

☐ Adalbert Ameise

☐ Carlo Chinchilla

☐ Anderer Spieler

☐ Keiner davon

1.3.2 Implementierung

Der folgende javascript Code wird dazu in den Fragetext (im Quellcode-Modus) eingefügt.

```
<script type="text/javascript" charset="utf-8">

$(document).on('ready pjax:scriptcomplete',function(){

    // The number of answers to be fixed at the end of the list
    var fixedAnswers = 1;

    // Set this to "true" if you want "Other" to be fixed in the last position
    var otherFixed = false;

    // Identify this question
    var qID = {QID};

    // Find the number of answers
    var ansCount = $('#question'+qID+' .answer-item').length;
    if($('#question'+qID+' input[type="text"]').length > 0) {
        ansCount = ansCount - 1
    }
    console.log(ansCount);

    // Place the last n answers created at the end of the list
    var fixedIndex = fixedAnswers - 1;
    for (var i=0; i<fixedAnswers; i++) {
```

```

    var answer = $('input[id^="answer"][id$="X'+qID+(ansCount-fixedIndex)+'"]');
    var answerItem = $(answer).closest('.answer-item');
    var answersList = $(answer).closest('ul');
    $(answersList).append(answerItem);
    fixedIndex--;
}

// Handle "Other"
if(otherFixed == true && $('#question'+qID+' input[type="text"]').length > 0) {
    var otherAnswer = $('#question'+qID+' input[type="text"]');
    var otherAnswerItem = $(otherAnswer).closest('.answer-item');
    var otherAnswersList = $(otherAnswer).closest('ul');
    $(otherAnswersList).append(otherAnswerItem);
}
});
</script>

```

Interessant sind die beiden Variablen „fixedAnswers“ und „otherFixed“.

„fixedAnswers“ bestimmt, wieviele der Teilfragen fest ans Ende gestellt werden.

„otherFixed“ bestimmt die Behandlung der „Sonstigen“-Option.

- „true“: betrachtet diese Option ebenfalls als zu fixierendes Element. Dann steht sie wieder am Schluss.
- „false“: Nur die anderen Optionen werden ans Ende fixiert.

Bemerkung: Dieses script funktioniert ohne Änderung auch für Einfachnennungsfragen

Feste Antworten am Ende einer Einfachnennung mit "Sonstigen"

☐ Berta Biber

☐ Doris Dachs

☐ Adalbert Ameise

☐ Carlo Chinchilla

☐ Anderer Spieler

☐ Sag ich nicht

☐ Weiß ich doch nicht

1.4 Beschränkung der anwählbaren Optionen

1.4.1 Anwendungsbeispiel

Um in Mehrfachnennungen nur eine bestimmte Anzahl auszuwählender Elemente zuzulassen, genügt es, die „Maximale Antwortzahl“ auf diesen Wert zu setzen.

D.h. wenn der Teilnehmer mehr als die erlaubte Zahl anwählt, erscheint eine Fehlermeldung.

Das ist gut und einsichtig.

Hier soll aber gezeigt werden, wie man nach Auswahl der erlaubten Anzahl die übrigen Kästchen einfach ausgraut und nicht mehr anwählbar macht.

Q1...

Check all that apply

- ☐ Adalbert Ameise
- ☒ Berta Biber
- ☐ Carlo Chinchilla
- ☒ Doris Dache
- ☐ Emil Eidechse
- ☒ Florian Flamingo
- ☐ Gabriele Gnu
- ☐ Hans Habicht
- ☐ Ilse Iltis
- ☐ Joachim Jaguar

Wenn dann z.B. ein Häkchen wieder entfernt wird, sind die anderen auch wieder anwählbar.

1.4.2 Implementierung

Der folgende javascript Code wird dazu in den Fragetext (im Quellcode-Modus) eingefügt.

```
<script type="text/javascript" data-author="Tony Partner">

$(document).on('ready pjax:scriptcomplete',function(){

    // Number of allowed answers
    var answers = 3;

    // Identify this question
    var thisQuestion = $('#question{QID}');

    // Listener on checkboxes
    $('.answer-item :checkbox', thisQuestion).on('change', function(e) {
        $('.answer-item :checkbox', thisQuestion).prop('disabled', false);
        if($('.answer-item :checkbox:checked', thisQuestion).length >= answers) {
            $('.answer-item :checkbox:not(:checked)', thisQuestion).prop('disabled', true);
        }
    });

    // Initial states
    if($('.answer-item :checkbox:checked', thisQuestion).length >= answers) {
        $('.answer-item :checkbox:not(:checked)', thisQuestion).prop('disabled', true);
    }
});

</script>
```

Am Anfang wird in der Variablen „answers“ angegeben, wie viele Antworten erlaubt sein sollen.

1.5 Fixe umfrageweite Randomisierung

1.5.1 Anwendungsbeispiel

Oftmals ist es in der Markt- und Meinungsforschung so, dass eine Anzahl von Items in zufälliger Reihenfolge dargeboten wird – um einen Bias möglichst auszuschließen – diese einmal gewählte Reihenfolge dann aber für die gesamte Umfrage beizubehalten.

Wenn also einmal die – zufällige – Reihenfolge „Biber“, „Dachs“, „Ameise“, „Chinchilla“, „Eidechse“ gezeigt wurde, soll auch in folgenden Fragen diese Reihenfolge erscheinen.

Ein anderes Vorgehen würde die Teilnehmer nur verwirren.

LimeSurvey bietet diese Möglichkeit standardmäßig nicht.

Hier wird in jeder Frage mit eingeschalteter Randomisierung eine neue Reihenfolge „ausgewürfelt“.

Aber es geht natürlich mit etwas javascript.

Dazu wird nach der ersten Frage, diese hat die Einstellung „Zufällige Reihenfolge: JA“, eine Frage vom Typ „kurzer freier Text“ erstellt.

Diese Frage wird in derselben Gruppe erzeugt und natürlich in der „scharfen“ Umfrage versteckt. Allerdings **nicht mit „Diese Frage immer verstecken“**, sondern mit der **css-Klasse „hidden“**.

In der Version 6.x. ist dies geändert in „d-none“.

Diese dient als Container, in welcher die initiale zufällige Reihenfolge gespeichert wird.

Mehrfachfrage 1

Bitte wählen Sie die zutreffenden Antworten aus:

- ☐ Carlo Chinchilla
- ☐ Adalbert Ameise
- ☐ Berta Biber
- ☐ Doris Dachs

Y003,Y001,Y002,Y004

1.5.2 Implementierung

In die Mehrfachfrage wird nun dieses kleine javascript-snippet (im Quellcode-Modus) eingefügt:

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:scriptcomplete',function(){

        //Identify the questions
        var thisQuestion = $('#question{QID}');
        var hiddenQuestion = $(thisQuestion).nextAll('.text-short:eq(0)');

        // Create an array of answer codes
        var answerCodes = [];
        $('li.answer-item', thisQuestion).each(function(i) {
            answerCodes.push($(this).attr('id').split('X{QID}')[1]);
        });

        // Load the hidden question
        $('input:text', hiddenQuestion).val(answerCodes);
    });
</script>
```

Nun kann man an jeder Stelle der Umfrage auf diese – versteckte – Frage zugreifen, um die dort gespeicherte Reihenfolge wieder zu reproduzieren.

Dazu wird dann in eine andere Frage dieses noch kleinere javascript eingefügt:

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:scriptcomplete',function(){

        //Identify this question
        var thisQuestion = $('#question{QID}');
        var thisAnswerList = $('li.answer-item:eq(0)', thisQuestion).parent();

        // Retrieve the answer codes from the "randomOrder" question
        var answerCodes = '{RandOrder1}'.split(',');

        // Loop through the answer codes
        $.each(answerCodes, function(i, val) {
            // Move the answer item
            $(thisAnswerList).append($('li.answer-item[id$="X{QID}'+val+'"]', thisQuestion));
        });
    });
</script>
```

Wichtig ist die Zeile `var answerCodes = '{RandOrder1}'.split(',');`

„RandOrder1“ ist der Fragencode der versteckten Frage, in welcher die Reihenfolge gespeichert ist. Dies muss dann gegebenenfalls angepasst werden.

Dann wird in einer zweiten Mehrfachnennungsfrage die ursprüngliche Reihenfolge wiederholt.
Diese Frage kann dann die Einstellung „Zufällige Reihenfolge: NEIN“ haben.

Mehrfachfrage 2

Bitte wählen Sie die zutreffenden Antworten aus:

☐ Carlo Chinchilla

☐ Adalbert Ameise

☐ Berta Biber

☐ Doris Dachs

Dies ist nun nicht die einzige Möglichkeit.

Man kann auch die Reihenfolge einer Mehrfachnennungsfrage später in einer Einfachnennungsfrage reproduzieren.

Dies ist ja auch ein oft benutztes Szenario. Zum Beispiel:

Welche dieser Automarken haben Sie schon besessen? (MFN)

Welche Marke fahren Sie zur Zeit? (EFN)

Hier würde zusätzlich der Matrixfilter eingesetzt werden, um nur noch die bereits gefahrenen Marken anzuzeigen.

Es muss man nur beachtet werden, dieselben Codes als Antwortoptionen zu benutzen wie bei den Teilfragen der MFN.

Einfach

Bitte wählen Sie eine der folgenden Antworten:

☐ Carlo Chinchilla

☐ Adalbert Ameise

☐ Berta Biber

☐ Doris Dachs

Desweiteren kann man diese Reihenfolge auch in einer folgenden Matrix wieder reproduzieren. Dazu benötigen wir aber ein etwas anderes script für die Matrixfrage.

```
<script type="text/javascript" data-author="Tony Partner">
  $(document).on('ready pjax:scriptcomplete',function(){

    //Identify this question
    var thisQuestion = $('#question{QID}');
    var thisAnswerList = $('tr.answers-list:eq(0)', thisQuestion).parent();

    // Retrieve the answer codes from the "randomOrder2" question
    var answerCodes = '{RandOrder1}'.split(',');

    // Loop through the answer codes
    $.each(answerCodes, function(i, val) {
      // Move the answer item
      $(thisAnswerList).append($('tr.answers-list[id$="X{QID}'+val+'"]', thisQuestion));
    });

  });
</script>
```

Matrix					
	sehr gut	gut	mittelmäßig	schlecht	sehr schlecht
Carlo Chinchilla	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Adalbert Ameise	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Berta Biber	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Doris Dachs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Um dies auch in einer Matrix(Texte) anzuwenden, benötigen wir wieder ein etwas anderes script. Nämlich:

```
<script type="text/javascript" data-author="Tony Partner">
  $(document).on('ready pjax:scriptcomplete',function(){

    //Identify this question
    var thisQuestion = $('#question{QID}');
    var thisAnswerList = $('tr.subquestion-list:eq(0)', thisQuestion).parent();

    // Retrieve the answer codes from the "RandOrder1" question
    var answerCodes = '{RandOrder1}'.split(',');

    // Loop through the answer codes
    $.each(answerCodes, function(i, val) {
      // Move the answer item
      $(thisAnswerList).append($('tr.subquestion-list[id$="X{QID}'+val+'"]', thisQuestion));
    });

  });
</script>
```

Matrix(Texte)				
	Sportart	Eintrittsdatum	Altersklasse	Bemerkungen
Carlo Chinchilla	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Adalbert Ameise	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Berta Biber	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Doris Dachs	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Wie man umgekehrt die Reihenfolge einer anfänglichen Matrix in einer Mehrfach- oder Einfachnennung wiederholt, wird im „Tutorial 1: Matrizen“ gezeigt.

Ich möchte ja, dass auch dieses gelesen wird.

2 Mehrfachnennung mit Kommentar

2.1 Ausgeblendete Kommentarfelder

2.1.1 Anwendungsbeispiel

Oft kommt es vor, dass man mehrere „Sonstige“ abfragen möchte.

Q2a

Kommentieren wenn eine Antwort gewählt wird

<input type="checkbox"/>	staubgrau	
<input type="checkbox"/>	steingrau	
<input type="checkbox"/>	mausgrau	
<input type="checkbox"/>	anderes grau, nämlich:	<input type="text"/>
<input type="checkbox"/>	karmesinrot	
<input type="checkbox"/>	purpurrot	
<input type="checkbox"/>	feuerrot	
<input type="checkbox"/>	anderes rot, nämlich:	<input type="text"/>
<input type="checkbox"/>	andere Farbe, nämlich:	<input type="text"/>
<input type="checkbox"/>	keine Angabe	

Standardmäßig bietet LimeSurvey nicht an, jeder Teilfrage ein Attribut „Sonstige“ hinzuzufügen, damit diese dann zusätzlich ein Textfeld anzeigt.

Hier muss man zum Fragetyp „Mehrfachauswahl mit Kommentar“ greifen.

Normalerweise werden zwar für jede Teilfrage offene Textfelder angezeigt; doch wir werden die nicht benötigten einfach entfernen.

2.1.2 Implementierung

In den Fragetext wird im Quellcode-Modus folgendes kleine javascript-snippet eingefügt.

```
<script type="text/javascript" charset="utf-8">
    $(document).ready(function() {
        var thisQuestion = $('#question{QID}');
        // Remove some text inputs
        $('.checkbox-text-item:eq(0) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(1) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(2) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(4) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(5) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(6) .comment-item', thisQuestion).remove();
        $('.checkbox-text-item:eq(9) .comment-item', thisQuestion).remove();
    });
</script>
```

Es werden also einfach die Zeilen aufgelistet, deren Textfeld entfernt werden soll.

Dabei ist zu beachten, dass die Zählung mit „0“ startet.

2.2 Geänderte Kommentarfeldlänge, Header

2.2.1 Anwendungsbeispiel

Man kann diese „Mehrfachnennung mit Kommentar“ auch dazu benutzen, sowohl das Vorhandensein von Objekten als auch deren Menge abzufragen.

Hier zum Beispiel eine Frage wie

Welche Farben benutzen Sie?

Geben Sie bei den benutzten Farben auch die prozentualen Anteil an!

Q2b

<input checked="" type="checkbox"/> grau	
<input type="checkbox"/> hellgrau	<input type="text"/> %
<input type="checkbox"/> steingrau	<input type="text"/> %
<input type="checkbox"/> anthrazitgrau	<input type="text"/> %
<input type="checkbox"/> zementgrau	<input type="text"/> %
<input checked="" type="checkbox"/> andere Farbe	
<input type="checkbox"/> gelb	<input type="text"/> %
<input type="checkbox"/> blau	<input type="text"/> %
<input type="checkbox"/> rot	<input type="text"/> %
<input type="checkbox"/> grün	<input type="text"/> %

Natürlich kann man dies auch in einer Matrix(Zahlen) abfragen mit voreingestelltem Summenwert von 100.

Um in einer Analyse aber auch die Häufigkeit der Farben berechnen zu können, müsste man zusätzliche Spalten generieren und berechnen.

Oder eine Mehrfachnennung vorschalten „Welche Farben ...“ und dann eine „mehrfache Zahleingabe“, die nur die benutzten Farben anzeigt.

Es gibt also auch andere Möglichkeiten. Hier sind nun zusätzlich noch Header eingebaut und ein Suffix hinter den Texteingabefeldern. Die Größe dieser Felder ist außerdem geändert.

2.2.2 Implementierung

Wie immer wird der javascript- und css-Code in den Fragetext (im Quellcode-Modus) eingefügt.

Zunächst die Header. Diese sind anders implementiert als in Kapitel 1.a..

Dort waren die Header ja richtige Teilfragen. Hier werden sie als separate Texte eingefügt.

```
<script type="text/javascript" charset="utf-8">
    $(document).ready(function() {
        var SubHeading1="grau";
        var SubHeading2="andere Farbe";
        var thisQuestion = $('#question{QID}');
        // Insert sub-headings
        $('#checkbox-text-item:eq(0)', thisQuestion).before('<li class="inserted-sub-heading"><span class="myHeader">'+SubHeading1+'</span></li>');
        $('#checkbox-text-item:eq(4)', thisQuestion).before('<li class="inserted-sub-heading"><span class="myHeader">'+SubHeading2+'</span></li>');
    });
</script>
```

Hier wird wieder die css-Klasse „myHeader“ benutzt, um diesen zu stylen.

```
<style type="text/css">.myHeader {
    color:maroon;
    font-size:120%;
    background-color:#F8F8FF;
    border:1px solid #ccc;
    padding: 1px 5px;
}
</style>
```

Nun kommt der javascript-Code für die Suffixe

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready ajax:scriptcomplete',function(){
        // Insert comment suffix
        var suffixText = '%';
        $('#question{QID} .comment-item').append('<span class="comment-suffix">'+suffixText+'</span>');
    });
</script>
```

Und zu guter Letzt kann man noch die Breite des Eingabefeldes definieren.

Wenn dieses Feld über die Eigenschaft „Texteingabebreite“ eingestellt wird, so wird es auf kleineren Eingabegeräten schmaler und schmaler, bis keine drei Ziffern mehr sichtbar sind. Außerdem würde das Suffix unterhalb des Feldes angezeigt.

Daher noch dieser css-Code (die Breite wird auf 8% gesetzt, aber mindestens 50px)

```
<style type="text/css">
    #question{QID} input[type="text"] {
        width:8%;
        min-width: 50px;
        display: inline-block;
    }
</style>
```

2.3 Mehrfachnennung mit Kommentar, mehrere Spalten

2.3.1 Anwendungsbeispiel

Im Gegensatz zur „normalen“ Mehrfachnennung gibt es bei diesem Fragetyp keine Möglichkeit, über die Einstellungen die Teilfragen auf mehrere Spalten zu verteilen.

Dies ist jedoch manchmal angenehm, wenn es viele Teilfragen gibt, und der erwartete Text in den Kommentaren nur kurz ist. Natürlich können auch längere Kommentare eingetragen werden, doch zeigt sich immer wieder, dass Teilnehmer verunsichert sind, wenn ihr geplanter Text nicht in das Feld zu passen scheint.

Es ist also so etwas gewünscht:

Q2c

<input type="checkbox"/> Adalbert Ameise	<input type="text"/>	<input type="checkbox"/> Berta Biber	<input type="text"/>
<input type="checkbox"/> Carlo Chinchilla	<input type="text"/>	<input type="checkbox"/> Doris Dachs	<input type="text"/>
<input type="checkbox"/> Emil Eidechse	<input type="text"/>	<input type="checkbox"/> Florian Flamingo	<input type="text"/>
<input type="checkbox"/> Gabriele Gnu	<input type="text"/>	<input type="checkbox"/> Horst Habicht	<input type="text"/>
<input type="checkbox"/> Inge Illtis	<input type="text"/>	<input type="checkbox"/> Joachim Jaguar	<input type="text"/>

2.3.2 Implementierung

Die gesamte Implementierung beruht nur auf css im Quellcode des Fragetextes.

```
<style type="text/css">
  @media only screen and (min-width: 768px) {
    #question{QID} li.checkbox-text-item {
      float: left;
      width: 50%;
      margin-right: 0;
      margin-left: 0;
    }

    #question{QID} .text-item {
      padding-right: 30px;
      padding-left: 0;
    }
  }
</style>
```

Durch die Setzung von 50% wird also jeder Teilfrage nur die Hälfte des zur Verfügung stehenden Raumes eingeräumt; dadurch wandert die zweite Teilfrage rechts neben die erste.

Im Gegensatz zur „normalen“ Mehrfachnennung sind die Teilfragen hier also waagrecht angeordnet.

Zu beachten ist, dass das Layout zerstört wird, wenn manche Texte mehrzeilig sind, andere nicht. Da sollte man mit der Einstellung „**Auswahl Spaltenbreite**“ einen Wert finden, der dies vermeidet.

Man kann auch hier Header einsetzen, wie im Kapitel zuvor beschrieben wurde. Hier sollte man beachten, dass Header nur vor „geraden“ Teilfragen gesetzt werden.

Q2c

Buchstaben A-D

<input type="checkbox"/> Adalbert Ameise	<input type="text"/>	<input type="checkbox"/> Berta Biber	<input type="text"/>
<input type="checkbox"/> Carlo Chinchilla	<input type="text"/>	<input type="checkbox"/> Doris Dachs	<input type="text"/>

Weitere Buchstaben

<input type="checkbox"/> Emil Eidechse	<input type="text"/>	<input type="checkbox"/> Florian Flamingo	<input type="text"/>
<input type="checkbox"/> Gabriele Gnu	<input type="text"/>	<input type="checkbox"/> Horst Habicht	<input type="text"/>
<input type="checkbox"/> Inge Iltis	<input type="text"/>	<input type="checkbox"/> Joachim Jaguar	<input type="text"/>

2.4 Textausrichtung

2.4.1 Anwendungsbeispiel

In einer Mehrfachnennungsfrage und auch in einer Mehrfachnennung mit Kommentar sind die Teilfragen normalerweise ordentlich linksbündig ausgerichtet.

(Ich demonstriere dies hier mit dem Standard-Theme „fruity“. In dem sonst zu Demonstrationszwecken benutzten Theme ist die Lösung bereits eingebaut.)

Q2d

<input type="checkbox"/> Adalbert Ameise	
<input type="checkbox"/> Berta Biber	
<input type="checkbox"/> Carlo Chinchilla	
<input type="checkbox"/> Doris Dachs	

Dies ist allerdings nur so lange der Fall, wie die Teilfragen einzeilig sind. Sobald ein Zeilenumbruch notwendig wird, ändert sich das Ganze zu

Q2d

<input type="checkbox"/> Adalbert Ameise	
<input type="checkbox"/> Berta Biber	
<input type="checkbox"/> Carlo Chinchilla	
<input type="checkbox"/> Doris Dachs	
<input type="checkbox"/> Emil Elefant. Das ist dieser Typ mit den großen Ohren (aber nur, wenn er aus Afrika kommt) und der langen Nase, im Volksmund auch Rüssel genannt	
<input type="checkbox"/> Florian Flamingo	
<input type="checkbox"/> Gabriele Gnu	

Q2d

<input type="checkbox"/> Adalbert Ameise	
<input type="checkbox"/> Berta Biber	
<input type="checkbox"/> Carlo Chinchilla	
<input type="checkbox"/> Doris Dachs	
<input type="checkbox"/> Emil Elefant. Das ist dieser Typ mit den großen Ohren (aber nur, wenn er aus Afrika kommt) und der langen Nase, im Volksmund auch Rüssel genannt	
<input type="checkbox"/> Florian Flamingo	
<input type="checkbox"/> Gabriele Gnu	

Das ist natürlich nicht besonders schön. Noch „interessanter“ sieht es aber aus, wenn man einige der Textfelder ausblendet (Kapitel 2.1.)

Dann hängt der Text völlig in der Luft.

2.4.2 Implementierung

Abhilfe bringt dieses kleine css-Stückchen, welches man in den Fragetext (im Quellcode-Modus) einfügt

```
<style type="text/css">
label.control-label {
    text-align: left !important;
}
</style>
```

Um diese Änderung nicht in jede Frage – wo nötig – einfügen zu müssen, kann man es auch (ohne die umschließenden `<style>...</style>` tags ans Ende der „custom.css“ schreiben.

Ergebnis:

Q2d

<input type="checkbox"/> Adalbert Ameise	<input type="text"/>
<input type="checkbox"/> Berta Biber	<input type="text"/>
<input type="checkbox"/> Carlo Chinchilla	<input type="text"/>
<input type="checkbox"/> Doris Dachs	<input type="text"/>
<input type="checkbox"/> Emil Elefant. Das ist dieser Typ mit den großen Ohren (aber nur, wenn er aus Afrika kommt) und der langen Nase, im Volksmund auch Rüssel genannt	<input type="text"/>
<input type="checkbox"/> Florian Flamingo	<input type="text"/>
<input type="checkbox"/> Gabriele Gnu	<input type="text"/>

2.5 Mehrfachnennung, mehrspaltig mit Header

2.5.1 Anwendungsbeispiel

In den vorherigen Kapiteln wurden bereits mehrere Möglichkeiten gezeigt, in Mehrfachnennungsfragen Header einzufügen, um die Teilfragen besser zu gliedern. Das war auch gut und schön, solange die Liste der auszuwählenden Objekte nicht allzu lang war – und die Lösung für das mehrspaltige Layout erforderte eine Zusatzbedingung.

Hier also eine Lösung die universaler ist.

Bitte, wählen Sie alle Tiere aus, die Sie in den letzten 6 Monaten gesehen haben!

Vögel

☐ Albatros☐ Bekassine☐ Condor☐ Drossel
☐ Emu☐ Flamingo☐ Gänsegeier☐ Habicht
☐ Ibis☐ Jagdfalke☐ Kasuar☐ Sonstige Vögel:

Säugetiere

☐ Affe☐ Biber☐ Chinchilla☐ Dachs
☐ Esel☐ Fuchs☐ Gepard☐ Hirsch
☐ Iltis☐ Jaguar☐ Koala☐ Lama
☐ Mungo☐ Sonstige Säugetiere:

Fische

☐ Aal☐ Blauwal☐ Clownfisch☐ Delfin
☐ Elritze☐ Flunder☐ Goldbrasse☐ Hai
☐ Igelfisch☐ Sonstige Fische:
☐ Sonstige Fische:

Hier werden die Teilfragen im Gegensatz zur Standard-Darstellung zeilenweise dargestellt. Außerdem funktioniert die Behandlung gefilterter Teilfragen.

2.5.2 Implementierung

In diesem Fall zieht das benutzte javascript seine Informationen sehr stark aus der Codierung der Teilfragen.

So gibt es spezielle Codes für die Header und auch spezielle Codes für die „Sonstigen“-Option.

Hier das javascript, welches wie üblich in den Fragetext im Quellcode-Modus eingefügt wird.

```
<script type="text/javascript" data-author="Tony Partner">

$(document).on('ready pjax:scriptcomplete',function(){

    // Identify this question
    var qID = '{QID}';
    var thisQuestion = $('#question'+qID);

    /***** HEADER ROWS *****/
    // Find the header rows
    // Hier wird gesetzt, dass die Codes der Header mit „0“ beginnen
    var headerRowPrefix = '0';
    var headerRows = $('li[id^="javatbd"]', thisQuestion).filter(function() {
        return $(this).attr('id').includes('X'+qID+headerRowPrefix+'');
    });

    // Loop through the header rows
    $(headerRows).each(function(i) {
        // Assign a class name
        $(this).addClass('inserted-header-row')
        // Move the label
        .prepend($('label.control-label:eq(0)', this))
        // Remove unnecessary elements
        .find('input, div').remove();
    });

    /***** ANSWERS INTO COLUMNS *****/
    // Number of columns to display (max 5)
    var columns = 4;
    thisQuestion.addClass('with-inserted-columns columns-'+columns+'');

    /***** REMOVE SOME TEXT INPUTS *****/

    // Find the rows to keep text inputs
    // Hier wird analog gesetzt, dass „Sonstigen“-Optionen mit „oth“ starten
    var textRowPrefix = 'oth';
    var textRows = $('li[id^="javatbd"]', thisQuestion).filter(function() {
        return $(this).attr('id').includes('X'+qID+textRowPrefix+'');
    });

    // Remove text inputs from all except those rows
    $('li[id^="javatbd"]', thisQuestion).not(textRows).find('div.text-item').remove();
});

</script>
```


Und hier das css

```
<style data-author="Tony Partner" type="text/css">li.inserted-header-row {
    float: none;
    clear: both;
    flex: 0 0 100%;
}

li.inserted-header-row label {
/* Hier wird das Layout der Überschrift; kann man alles nach Gusto ändern */
    padding-left: 15px;
    padding-right: 15px;
    padding-top: 0 !important;
    font-weight: bold;
    width: 100%;
    background-color: #e29514;;
    border: 1px solid gray;
    border-radius: 5px;
    font-size: 18px;
    color: #001957;
}
li.inserted-header-row label::before,
li.inserted-header-row label::after{
    display: none;
}

@media only screen and (min-width: 768px) {
    .with-inserted-columns ul.ls-answers {
        display: flex;
        flex-flow: row wrap;
    }

    .with-inserted-columns ul.ls-answers li {
        margin: 0 0 1em 0;
        padding: 0 15px 0 0;
    }

    // Hier sieht man, warum die maximale Spaltenzahl auf 5 festgelegt ist;
    // man kann natürlich weitere Breiten hinzufügen, also 16%, 14%, ...
    // Es hängt von der Länge der Texte ab, ob daraus noch ein vernünftiges Aussehen resultiert
    .with-inserted-columns.columns-2 ul.ls-answers li { flex: 0 0 50%; }
    .with-inserted-columns.columns-3 ul.ls-answers li { flex: 0 0 33%; }
    .with-inserted-columns.columns-4 ul.ls-answers li { flex: 0 0 25%; }
    .with-inserted-columns.columns-5 ul.ls-answers li { flex: 0 0 20%; }

    .with-inserted-columns ul.ls-answers li.inserted-header-row {
        flex: 0 0 100%;
        padding: 0;
    }

    .with-inserted-columns ul.ls-answers li > div {
        float: none;
    }
}
```

```
        width: auto;
    }

    // Hier wird der Text des Headers zentriert
    .with-inserted-columns ul.ls-answers li label {
        text-align: center;
    }

    .with-inserted-columns ul.ls-answers li input[type="text"] {
        margin-top: 0.7em;
    }
}
</style>
```

3 Mehrfache kurze Texteingabe

3.1 Mit Drop-Down und Datepicker

3.1.1 Anwendungsbeispiel

Im Tutorial über Matrizen haben wir dies auch schon gesehen. Man kann diese dort gezeigte Lösung auch anwenden wenn es nur eine Zeile (ein Objekt) gibt, zu dem Daten abgefragt werden.


Ungünstig wird es allerdings, wenn es sich um viele Daten handelt; dann wird es etwas kanpp mit der Zeilenbreite.

Daher nimmt man in einem solchen Fall den Fragetyp „Mehrfache kurze Texte“.

Q3a

Name:

Vorname:

Eintrittsdatum: 

Abteilung:

<

November 2019

>

Su	Mo	Tu	We	Th	Fr	Sa
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
1	2	3	4	5	6	7

E-Mail-Adresse:

Name der Katze:


Schuhgröße:

Marke des Fahrrads:

Q3a

Name:

Vorname:

Eintrittsdatum: 

Abteilung:

--Bitte, Abteilung auswählen--

Fußball
Handball
Karate
Schwimmen
Volleyball
Andere Abteilung

E-Mail-Adresse:

Name der Katze:

Schuhgröße:


Marke des Fahrrads:

Hier haben wir also einen Datepicker und ein Drop-Down.

Zusätzlich öffnet sich ein weiteres Feld „Andere Abteilung“, wenn dies im Drop-Down gewählt wurde.

Dies ist natürlich über Teilfragenrelevanz gelöst, also „Q3a_SQ004==6“, wobei 6 der Code der Nennung „Andere Abteilung“ im Drop-Down ist.

Q3a

Name:	<input type="text"/>
Vorname:	<input type="text"/>
Eintrittsdatum:	<input type="text" value="12.11.2019"/> 
Abteilung:	<div>Andere Abteilung ▼</div>
Andere Abteilung	<input type="text"/>
E-Mail-Adresse:	<input type="text"/>
Name der Katze:	<input type="text"/>
Schuhgröße:	<input type="text"/>
Marke des Fahrrads:	<input type="text"/>

3.1.2 Implementierung

Wie immer wird der javascript-Code in den Fragetext (im Quellcode-Modus) eingefügt.

Zunächst das Drop-Down

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:complete',function() {
        var qID = {QID};
        var inputNum = 4;

        // Define the select element (dropdown)
        var prov1 = '<select id="prov1" class="form-control">\
            <option value="">-Bitte, Abteilung auswählen--</option>\
            <option value="1">Fußball</option>\
            <option value="2">Handball</option>\
            <option value="3">Karate</option>\
            <option value="4">Schwimmen</option>\
            <option value="5">Volleyball</option>\
            <option value="6">Andere Abteilung</option>\
        </select>';

        // Hide the text input
        $('#question'+qID+' .question-item:eq('+inputNum-1+') input[type="text"]').hide();

        // Insert the select elements
        if($('#question'+qID+' .question-item:eq('+inputNum-1+') select').length == 0) {
            $('#question'+qID+' .question-item:eq('+inputNum-1+')
            input[type="text"]').before(prov1);
        }
    })
</script>
```

```

// Initially select an option if the question has already been answered
$('#question'+qID+' select').each(function(i) {
    if($.trim($(this).next('input[type="text"]').val()) != '') {
        $(this).val($.trim($(this).next('input[type="text"]').val()));
    }
});

// Listener on the dropdowns - insert selected values into hidden text input
$('#question'+qID+' select').change(function() {
    var thisInput = $(this).next('input[type="text"]');
    $(thisInput).val($(this).val());
    checkconditions($(thisInput).attr('value'), $(thisInput).attr('name'), 'text');
});

// Some styles
$('#question'+qID+' select').css({
    'margin':'0.3em 0 0 0'
});
});
</script>

```

In der Zeile „var inputnum=4;“ wird bestimmt, in welcher Zeile das Drop-Down eingefügt wird. Diesmal beginnt man bei „1“ an zu zählen, da später im Code die „1“ wieder abgezogen wird.

Danach folgt das script für den Datepicker.

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:scriptcomplete',function(){
        var rootPath = location.pathname.split('index.php')[0];

        // Identify this question
        var thisQuestion = $('#question{QID}');

        $('head').append('<link rel="stylesheet"
href="'+rootPath+'assets/packages/bootstrap/plugins/datetimepicker/build/css/bootstrap-
datetimepicker.min.css" type="text/css" />');

        $.getScript(rootPath+'assets/packages/bootstrap/plugins/datetimepicker/build/js/bootstrap-
datetimepicker.min.js')
            .done(function(script, textStatus) {

                // Insert the date-time-pickers
                $('.answer-item:nth-child(3) input:text', thisQuestion).each(function(i) {

                    $(this).addClass('date-control date datetimepicker')
                        .wrap('<div class="inserted-date-wrapper input-group date date-timepicker-
group" />')
                        .after('<div class="input-group-addon datetimepicker-addon btn btn-primary">\
                            <i class="fa fa-calendar" aria-hidden="true"></i><span
class="sr-only"></span>\
                            </div>');

                    $(this).datetimepicker({
                        widgetParent: $(this).parent(),
                        useCurrent:false,
                        allowInputToggle: true,
                        format: 'DD.MM.YYYY'
                    });
                });
            })
            .fail(function( jqxhr, settings, exception ) {
                console.log(exception);
            });
    });
</script>
```

3.2 autocomplete

3.2.1 Anwendungsbeispiel

Zunächst eine Begriffserklärung. Unter „autocomplete“ verstehen wir, dass nach Eingabe einiger Zeichen alle Antwortmöglichkeiten eingeblendet werden, die diese Zeichenfolge enthalten und die sich bei Eingabe weiterer Zeichen aktualisiert. Dann kann man aus diesen das Gesuchte auswählen.

The image shows two side-by-side screenshots of a web form titled 'Q3b'. The form has three input fields: 'Name:', 'Wohnort:', and 'Geburtsland:'. In both screenshots, the 'Geburtsland:' field is active, and a dropdown menu is displayed below it. The dropdown menu lists various countries. In the left screenshot, the text 'Do' is entered in the input field, and the dropdown lists: Andorra, Barbados, Dominica, Dominikanische Republik, Ecuador, El Salvador, Indonesien, Mazedonien, and Südossetien. In the right screenshot, the text 'Dor' is entered, and the dropdown lists: Andorra, Ecuador, and El Salvador. The dropdown menu is highlighted with a blue border.

Zunächst ist dies ähnlich einem normalen Drop-Down.

Doch hier soll es um wirklich lange Listen gehen (mit vielleicht mehr als 1000 Elementen). Da stößt das normale Drop-Down an seine Grenzen.

Diese Liste kann man nun auf zwei Arten auslesen.

Man kann ein einfaches Array in den javascript-Code einfügen oder die Objekte aus einer Textdatei lesen.

Beispiel für ein Array:

```
["Adalbert Ameise", "Berta Biber", "Carlo Chinchilla", "Doris Dachs", "Ernst Eidechse", "Florian Flamingo", "Gabriele Gnu", „Horst Habicht“]
```

Beispiel für eine Textdatei (pro Zeile ein Element):

```
Adalbert Ameise
Berta Biber
Carlo Chinchilla
Doris Dachs
Ernst Eidechse
Florian Flamingo
Gabriele Gnu
Horst Habicht
```

3.2.2 Implementierung

3.2.2.1. als Matrix

In den Fragetext (im Quellcode-Modus) wird dieser javascript-Code eingefügt

Zunächst benötigen wir einige zusätzliche Bibliotheken. Diese werden hier direkt aus dem Internet geladen aus dem CDN von „jquery“ (Content Delivery Network)

```
<link href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css" rel="stylesheet" />
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
```

Danach folgt der eigentliche Code für unser autocomplete.

```
<script type="text/javascript" charset="utf-8">
  $(document).ready(function() {
    $('#question{QID} input[type="text"]:eq(2)').autocomplete({
      minLength: 1,
      source: ["Adalbert Ameise", "Berta Biber", "Carlo Chinchilla", "Doris Dachs", "Ernst Eidechse", "Florian Flamingo", "Gabriele Gnu",
"Horst Habicht"]
    });
  });
</script>
```

Wichtig ist hier die Zeile

`$('#question{QID} input[type="text"]:eq(2)').autocomplete({`

„eq(2)“ heißt, dass das „autocomplete“-Element in der dritten Zeile eingefügt wird (die Zählung beginnt bei 0 für die erste Zeile)

Und mit

`minLength: 1,`

passt man an, ab dem wievielten Buchstaben das „autocomplete“ anspringen soll (hier also gleich nach Eingabe des ersten Buchstabens)

3.2.2.2. als Textdatei

Zusätzlich zu den oben genannten beiden Bibliotheken wird hier noch eine dritte benötigt. Diese liest die Daten aus der Datei.

Also zunächst die drei Dateien

```
// Die zwei bekannten
<link href="//code.jquery.com/ui/1.12.1/themes/base/jquery-ui.css" rel="stylesheet" />
<script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
// Und diese hier zusätzlich
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-csv/1.0.11/jquery.csv.min.js"></script>
```

Danach folgt der eigentliche Code zum Aufruf

```
<script type="text/javascript" charset="utf-8">
  $(document).on('ready pjax:complete',function() {
    // Hier wird der Speicherort der Datei angegeben
    var url = "/upload/surveys/{SID}/files/namen.csv";

    var Names = new Array();

    $.get(url,function(data){
      fullArray = $.csv.toArrays(data);
      $(fullArray).each(function(i, item){
        Names.push(item[0]);
      });
      $("#question{QID} input[type=text]:eq(0)").autocomplete({
        minLength: 2,
        source: Names
      });
    });
  });
</script>
```

Die Bezeichnung (hier „Names“) ist beliebig wählbar; sie muss aber an den drei markierten Stellen identisch sein.

In der im Anhang befindlichen Beispiel-Datei ist als Beispiel der obige screenshot genommen; der Name stammt aus einer Textdatei, ebenso PLZ und Wohnort (mit mehr als 9000 Einträgen), das Geburtsland stammt aus einer Matrix.

Die beiden Textdateien müssen also ins „files“-Verzeichnis der Umfrage hochgeladen werden.

Sie sind als zip im Anhang, ebenso die Bibliotheken, die oben aus dem CDN geladen wurden. DSGVO-konform sollten diese ebenfalls in das „files“-Verzeichnis kopiert und von dort aufgerufen werden.

Beachte:

In der Beispiel-Iss werden diese Dateien von meinem Server geladen.
Dies kann manchmal auch nicht funktionieren.

Diese Implementierung eines „autocomplete“ funktioniert ebenso in einer Matrix(Texte).

Dazu wird nur die Zeile (beachte: „eq(x)“ gibt die Zeile an)

```
$("#question{QID} input[type=text]:eq(0)").autocomplete({
```

durch diese ersetzt (beachte: „:nth-child(x)“ gibt die Spalte der Matrix an, Spalte der Teilfragen zählt mit).

```
$('#question{QID} .answer-item:nth-child(2) input[type="text"]').autocomplete({
```

3.2.3 Was könnte dagegen sprechen

So schön dies auch ist, man bekommt nie die gesamte Liste zu sehen.

Nehmen wir als Beispiel die Suche nach der jungen Dame mit dem Namen „Ysabel Amanda Guerrero Herrera“.

Wenn nun der Umfrageteilnehmer den Namen nur gehört hat und daher denkt, der Vorname sei „Isabel“ oder „Isabelle“ und daher „Is“ eingibt, wird er nie ein Ergebnis bekommen. Bis ihm dann vielleicht einfällt, dass es auf Kuba und auch in der Dominikanischen Republik lange Zeit Mode war, jeden Namen möglichst mit „Y“ anfangen zu lassen (sieht man sehr gut bei Sportübertragungen).

Oder er sucht nach „Gu“ oder „He“.

3.3 Autocomplete mit Bibliothek „easy-autocomplete“

3.3.1 Anwendungsbeispiel

Im Gegensatz zum vorigen Beispiel, bei dem die Drop-Downs ja unabhängig voneinander waren, kann man mit dieser Bibliothek auch so etwas wie eine Baum-Struktur kreieren.

Etwas Ähnliches gibt es auch als Darstellung in einer Dual Matrix im „Tutorial 1: Matrizen“, Kap. 15.

Hier nun kann man aber auch mehr als zweistufig anzeigen.

Hier nun ebenfalls das Beispiel „Bundesländer – Landkreise“ in Deutschland.

3.4 Verschiedene Suffixe der Teilfragen

3.4.1 Anwendungsbeispiel

Stellen wir uns vor, wir wollen einen Test machen, der das Wissen über verschiedene Maßeinheiten abfragt.

Q3c

70° Celsius =	<input type="text"/>	° Fahrenheit
34° Reaumur =	<input type="text"/>	° Kelvin
145° Fahrenheit =	<input type="text"/>	° Reaumur
17° Kelvin =	<input type="text"/>	° Celsius

Zugegeben, dieses Beispiel würde man eher in einer Matrix(Zahlen) anlegen, aber man sieht, wie es funktionieren soll.

3.4.2 Implementierung

Sehr wichtig ist, dass in „Anzeige“ irgendein Dummy-Suffix eingetragen wird.

Danach wird (wie immer im Quellcode der Frage) dieses javascript-Stückchen eingetragen.

```
<script type="text/javascript" data-author="Tony Partner">
  $(document).on('ready pjax:scriptcomplete',function(){

    // Change the first suffix
    $('#question{QID} .suffix-text:eq(0)').html('° Fahrenheit');

    // Change the second suffix°
    $('#question{QID} .suffix-text:eq(1)').html('° Kelvin');

    // Change the third suffix
    $('#question{QID} .suffix-text:eq(2)').html('° Reaumur');

    // Change the fourth suffix
    $('#question{QID} .suffix-text:eq(3)').html('° Celsius');

  });
</script>
```

Durch die Benutzung von „.html“ ist es möglich den Text auch noch zusätzlich mit einem `Mein Text` zu formatieren.

3.5 „Dynamische“ Anzeige der Eingabefelder

3.5.1 Anwendungsbeispiel

Dies ist das Analogon zu Kapitel 6. im „[Tutorial 1: Matrizen](#)“.

Daher sei auch dorthin verwiesen.

Bitte nennen bis zu fünf Ihrer Lieblingsopern!

1.

Bitte nennen bis zu fünf Ihrer Lieblingsopern!

1.

2.

Und wenn die Maximalzahl errericht ist, gibt es nur noch den „Löschen“-Button.

Bitte nennen bis zu fünf Ihrer Lieblingsopern!

1.

2.

3.

4.

5.

3.5.2 Implementierung

Für das script könnte ich auch sagen „Es ist identisch mit dem script für Matrizen, nur an 4 Stellen muss ein Wort ausgetauscht werden“.

Schließlich handelt es sich bei dieser Frage nicht um eine Tabelle, sondern um eine Liste.

Daher

Zweimal hier

Von

```
// Move them to after the array
$( 'div#addButton'+qID ).appendTo( $( '#question' + qID + ' table.ls-answers' ).parent());
$( 'div#removeButton'+qID ).appendTo( $( '#question' + qID + ' table.ls-answers' ).parent());
```

Nach

```
// Move them to after the array
$( 'div#addButton'+qID ).appendTo( $( '#question' + qID + ' ul.ls-answers' ).parent());
$( 'div#removeButton'+qID ).appendTo( $( '#question' + qID + ' ul.ls-answers' ).parent());
```

und ebenfalls hier

Von

```
// Function to add a row, also shows the Remove element and hides the
//Add element if all rows are shown
function addRow(qID) {
    var arrayRow = '#question' + qID + ' table.ls-answers tr.subquestion-list';
    ....
}
```

Nach

```
// Function to add a row, also shows the Remove element and hides the
//Add element if all rows are shown
function addRow(qID) {
    var arrayRow = '#question' + qID + ' ul.ls-answers li.answer-item';
    ....
}
```

Und dasselbe in **function removeRow(qID)**

Trotzdem hier noch einmal das gesamte script.

```
<script>
$(document).ready(function() {

    // A function to add or remove rows of an "multiple short text" question
    function varLengthArray(qID) {

        if ($('#question'+qID+'.').length > 0) {

            // The HTML content of the Add/Remove elements - modify as you wish
            var addContent = '[+] Zeile zufügen';
            var removeContent = '[-] Zeile löschen';

            // Create the Add and Remove elements & insert them
            // Adjust colors by using other bootstrap classes
            // „btn-primary“, „btn-success“, „btn-info“, „btn-warning“, „btn-danger“
            var el1 = document.createElement('div');
            el1.setAttribute('id','addButton'+qID);
            el1.setAttribute('class','btn btn-success');
            document.body.appendChild(el1);
            var el2 = document.createElement('div');
            el2.setAttribute('id','removeButton'+qID);
            el2.setAttribute('class','btn btn-danger');
            document.body.appendChild(el2);

            // Move them to after the array
            $( 'div#addButton'+qID ).appendTo( $( '#question' + qID + ' ul.ls-answers' ).parent());
            $( 'div#removeButton'+qID ).appendTo( $( '#question' + qID + ' ul.ls-answers' ).parent());

            // Insert their HTML
            $( 'div#addButton'+qID ).html( addContent );
            $( 'div#removeButton'+qID ).html( removeContent );

            // Style the elements - you can modify here if you wish
            $( 'div#addButton'+qID ).css({
                'margin':'10px 0 10px 10px',
                'padding':'1px',
                'text-align':'center',
                'width':'auto',
                'cursor':'pointer',
                'float':'left'
            });

            $( 'div#removeButton'+qID ).css({
                'margin':'10px 0 10px 10px',
                'padding':'1px',
                'text-align':'center',
                'width':'auto',
                'cursor':'pointer',
                'float':'left'
            });
        }
    }
});
```

```

// Initially hide the Remove element
$( 'div#removeButton'+qID ).hide();

// Call the functions below when clicked
$( 'div#addButton'+qID ).click(function (event) {
    addRow(qID);
});
$( 'div#removeButton'+qID ).click(function (event) {
    removeRow(qID);
});

// Function to add a row, also shows the Remove element and hides the
//Add element if all rows are shown
function addRow(qID) {
    var arrayRow = '#question' + qID + ' ul.ls-answers li.answer-item';
    var rowCount = $( arrayRow ).size() - 1;
    $( arrayRow + '[name="hidden"]:first' ).attr('name', 'visible').show();
    $( 'div#removeButton'+qID ).show();
    if ( $( arrayRow + ':eq(' + rowCount + ')') .attr('name') == 'visible' ) {
        $( 'div#addButton'+qID ).hide();
    }
}

// Function to remove a row, also clears the contents of the removed row,
// shows the Add element if the last row is hidden and hides the Remove
// element if only the first row is shown
function removeRow(qID) {
    var arrayRow = '#question' + qID + ' ul.ls-answers li.answer-item';
    var rowCount = $( arrayRow ).size() - 1;
    $( arrayRow + '[name="visible"]:last input[type="text"]' ).val('');
    $( arrayRow + '[name="visible"]:last' ).attr('name', 'hidden').hide();
    $( 'div#addButton'+qID ).show();
    if ( $( arrayRow + ':eq(1)' ).attr('name') == 'hidden' ) {
        $( 'div#removeButton'+qID ).hide();
    }
}

// Just some initialization stuff
var arrayRow = '#question' + qID + ' ul.ls-answers li.answer-item';
var rowCount = '';

// Initially hide all except first row or any rows with populated inputs
$( arrayRow ).each(function(i) {
    if ( i > 0 ) {
        // We also need to give the hidden rows a name cause IE doesn't
        // recognize jQuery :visible selector consistently
        $( this ).attr('name', 'hidden').hide();

        $('input[type=text]', this).each(function(i) {
            if ($(this).attr('value') != '') {

```



```
        $(this).parents('tbody:eq(0)').attr('name', 'visible').show();
        $( 'div#removeButton'+qID ).show();
    }
    });
    rowCount = i;
    }
    });
    }
}
// Call the function with a question ID
varLengthArray({QID});
});
</script>
```

Anpassen kann man überall dort, wo es auch geschrieben steht, also den Text der Buttons, die bootstrap-Klasse und weiter unten in den beiden „css“-Sektionen.

4 Slider

4.1 Slider mit Ticks und Bildern/Smileys

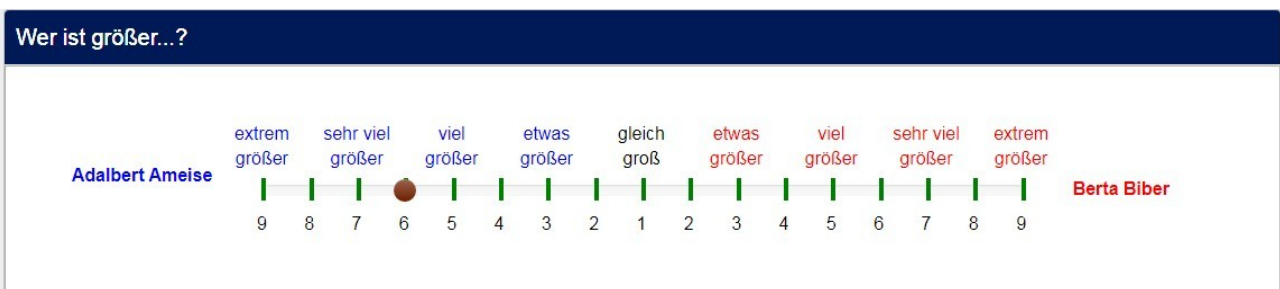
4.1.1 Anwendungsbeispiele

Dies ist eine Möglichkeit, einen Slider ein wenig „aufzumotzen“. Aber natürlich bietet es auch den Vorteil, Skalen unterschiedlich von einer aufsteigenden 0-n-Skala darzustellen.

Sieht dann zunächst so aus, mit dem Handle in der Mitte



Da der gewohnte Balken bis zum Handle bei dieser Art der Skala verwirren würde, wird nur das Handle zum gewünschten Wert verschoben



Hier ein Beispiel mit Smileys. Zunächst wird gar kein Handle gezeigt.



Erst, wenn geklickt wird, wird das Handle sichtbar und kann dann verschoben werden.



4.1.2 Implementierung

Dies ist sowohl ein langes javascript und auch ein langes css. Diese beruhen wieder auf einer Arbeit vom Kollegen Tony Partner (tpartner).

Zum ersten Beispiel:

Im Grunde ist dies ja ein Slider von 1-17; ihm werden nur andere Beschriftungen „untergejubelt“.

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready ajax:scriptcomplete',function(){
        var ticksArray = [
            [1, '9'],
            [2, '8'],
            [3, '7'],
            [4, '6'],
            [5, '5'],
            [6, '4'],
            [7, '3'],
            [8, '2'],
            [9, '1'],
            [10, '2'],
            [11, '3'],
            [12, '4'],
            [13, '5'],
            [14, '6'],
            [15, '7'],
            [16, '8'],
            [17, '9']
        ];
        var ticksArray2 = [
            [1, '<span style="color:blue">extrem<br>größer</span>'],
            [2, ''],
            [3, '<span style="color:blue">sehr viel größer</span>'],
            [4, ''],
            [5, '<span style="color:blue">viel <br> größer</span>'],
            [6, ''],
            [7, '<span style="color:blue">etwas <br> größer</span>'],
            [8, ''],
            [9, '<span style="color:black">gleich <br> groß</span>'],
            [10, ''],
            [11, '<span style="color:red">etwas <br> größer</span>'],
            [12, ''],
            [13, '<span style="color:red">viel <br> größer</span>'],
            [14, ''],
            [15, '<span style="color:red">sehr viel größer</span>'],
            [16, ''],
            [17, '<span style="color:red">extrem<br>größer</span>']
        ];

        insertSliderTicks('{QID}', ticksArray);
        insertSliderTicks2('{QID}', ticksArray2);
    });
```

```

/* Insert Slider Tick Marks */
function insertSliderTicks(qID, ticksArray) {
    var thisQuestion = $('#question'+qID);

    $(thisQuestion).addClass('with-inserted-ticks');

    $('input:text', thisQuestion).on('slideEnabled',function(){
        var thisSlider = $(this);
        var thisItem = $(thisSlider).closest('li');
        var thisMin = $('.slider-handle:eq(0)', thisItem).attr('aria-valuemin');
        var thisMax = $('.slider-handle:eq(0)', thisItem).attr('aria-valuemax');
        var thisRange = thisMax - thisMin;

        $.each(ticksArray, function(i, val) {
            var tickRelativePosition = val[0] - thisMin;
            var tickPercent = (tickRelativePosition/thisRange)*100;

            // Insert tick marks
            $('.slider-handle:eq(0)', thisItem).before('<div class="inserted-tick
left-'+tickPercent+'" style="left: '+tickPercent+'%">\
                <div class="tick-text">'+val[1]+'</div>\
            </div>');

        });
    });
}

/* Insert Slider Tick Marks */
function insertSliderTicks2(qID, ticksArray2) {
    var thisQuestion = $('#question'+qID);

    $(thisQuestion).addClass('with-inserted-ticks');

    $('input:text', thisQuestion).on('slideEnabled',function(){
        var thisSlider = $(this);
        var thisItem = $(thisSlider).closest('li');
        var thisMin = $('.slider-handle:eq(0)', thisItem).attr('aria-valuemin');
        var thisMax = $('.slider-handle:eq(0)', thisItem).attr('aria-valuemax');
        var thisRange = thisMax - thisMin;
        $.each(ticksArray2, function(i, val) {
            var tickRelativePosition = val[0] - thisMin;
            var tickPercent = (tickRelativePosition/thisRange)*100;

            // Insert tick marks
            $('.slider-handle:eq(0)', thisItem).before('<div class="inserted-tick2
left-'+tickPercent+'" style="left: '+tickPercent+'%">\
                <div class="tick-text">'+val[1]+'</div>\
            </div>');

        });
    });
}
</script>

```

Und hier der css-Teil:

Hier muss recht viel angepasst werden, damit das Resultat auch den Wünschen entspricht.

Dies betrifft die Farbgebung, die Abstände, das Alignment, usw.

```
<style type="text/css">/* Slider Tick Marks */

@media only screen and (min-width: 768px) {
    .slider-list .slider-left, .slider-list .slider-right { margin-top: 0.5em; }
    .with-inserted-ticks .slider-container {
        padding-right: 30px;
        padding-left: 30px;
    }
}

.with-inserted-ticks .slider-item { margin-bottom: 50px; }

.with-inserted-ticks .slider-container .help-block {
    margin: 25px 0 0 -20px;
    width: 40px;
    text-align: center;
}

.with-inserted-ticks .slider-container .help-block.pull-right {
    margin: 25px -20px 0 0;
}

.inserted-tick, .inserted-tick2 {
    position: absolute;
    top: 20%;
    height: 20px;
    width: 4px;
    margin-top: -5px;
    margin-left: -1px;
    background-color: green;
}

.inserted-tick2.left-0,
.inserted-tick2.left-100,
.inserted-tick.left-0,
.inserted-tick.left-100 {
    background-color: green;
}

.inserted-tick2 .tick-text {
    position: absolute;
    top: -300%;
    width: 100px;
    margin-left: -50px;
    color: #000000;
    text-align: center;
}
```

```
.inserted-tick .tick-text {
  position: absolute;
  top: 150%;
  width: 100px;
  margin-left: -50px;
  color: #000000;
  text-align: center;
}

@media only screen and (max-width: 768px) {

  .inserted-tick.left-0 .tick-text {
    margin-left: 0px;
    text-align: left;
  }

  .inserted-tick.left-100 .tick-text {
    margin-left: -100px;
    text-align: right;
  }

}

.ls-answers { padding-top:75px; }
.slider .tooltip { display:none !important; }
.slider-selection { display:none; }
</style>
```

Der javascript-Teil des zweiten Beispiels ist analog.

Wir haben es hier mit einem Slider von 1-15 zu tun und haben ein paar Smileys.

Daher nur der erste Teil, der die „ticksarrays“ definiert.

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:scriptcomplete',function(){

        var ticksArray = [
            [1, '-7'],
            [2, '-6'],
            [3, '-5'],
            [4, '-4'],
            [5, '-3'],
            [6, '-2'],
            [7, '-1'],
            [8, '0'],
            [9, '1'],
            [10, '2'],
            [11, '3'],
            [12, '4'],
            [13, '5'],
            [14, '6'],
            [15, '7'],
        ];
        var ticksArray2 = [
            [1, ''],
            [2, ''],
            [3, ''],
            [4, ''],
            [5, ''],
            [6, ''],
            [7, ''],
            [8, ''],
            [9, ''],
            [10, ''],
            [11, ''],
            [12, ''],
            [13, ''],
            [14, ''],
            [15, ''],
        ];

        insertSliderTicks('{QID}', ticksArray);
        insertSliderTicks2('{QID}', ticksArray2);
    });
... usw.
```

Der css-Teil ist ebenfalls nahezu identisch. Nur einige Abstände müssen angepasst werden.

Zum Beispiel steht im ersten Beispiel:

```
.slider-list .slider-left, .slider-list .slider-right { margin-top: 0.5em; }
```

D.h. für den Text der linken und der rechten Seite wird derselbe Abstand genommen.

Im Gegensatz dazu das zweite Beispiel, in welchem die linke Seite ja zweizeilig ist

```
.slider-list .slider-left { margin-top: -0.25em; }  
.slider-list .slider-right { margin-top: 0.5em; }
```

Da ja das Handle beim Start nicht angezeigt werden soll, kommt diese Zeile noch hinzu.

```
.slider.slider-untouched .slider-handle { display: none; }
```


4.2 Slider mit wechselnden Tooltips

4.2.1 Anwendungsbeispiel

In den vorigen Beispielen wurde der Tooltip des Sliders verborgen. Hier soll er gezeigt und auch gleich mit etwas Leben gefüllt werden. Nämlich, indem im Tooltip nun eigene Texte angezeigt werden, die die Skala beschreiben.



Oder man stellt den einzugebenden Zahlwert ein bisschen „schöner“ dar.



Dies würde man auch mit den vorher gezeigten „Ticks“ hinbekommen; allerdings kann es passieren, dass einem einfach „der Platz ausgeht“, und das Layout nicht mehr ansprechend aussieht.

4.2.2 Implementierung

Auch hier wird wieder ein kleines javascript-snippet in den Quellcode des Fragetextes eingefügt.

```
<script type="text/javascript" charset="utf-8">

$(document).on('ready pjax:scriptcomplete',function(){

    // Identify this question
    var thisQuestion = $('#question{QID}');

    // Define the text strings
    var tipTexts = {
        1: 'Super toll',
        2: 'Zur vollsten Zufriedenheit',
        3: 'Im Rahmen der Fähigkeiten',
        4: 'Nicht ausreichend',
        5: 'Katastrophal'
    };

    $('input:text', thisQuestion).on('slideEnabled',function(){
        var thisItem = $(this).closest('li');

        // Insert custom tooltip
        $('.tooltip-inner', thisItem).addClass('tooltip-inner-1 hidden');
        $('.tooltip', thisItem).append('<div class="tooltip-inner tooltip-inner-2">'+tipTexts[$
(this).val()]+</div>');

        // Listener on slider
        $(this).on('slide slideStop', function(event) {
            // Handle dynamic tooltip text
            $('.tooltip-inner-2', thisItem).text(tipTexts[$(this).val()]);
        });
    });
});

</script>
```

Mithilfe eines kleinen css-Teils wird nun in der ersten Zeile der Tooltip versteckt, wenn der Slider noch nicht „angefasst“ wurde.

Danach folgt ein beliebiges Styling des Tooltips

```
<style type="text/css">
.slider.slider-untouched .tooltip { display:none; }
.tooltip-inner-2 {
    background-color: yellow; /* Hintergrundfarbe „gelb“ */
    color: maroon;           /* Schriftfarbe „maroon“ */
    font-weight: bold;       /* Schriftauszeichnung „fett“ */
    border: 1px solid blue;   /* Rahmen „1px durchgezogen blau“ */
    font-size: 14px;         /* Schriftgröße „14 pixel“ */
    padding: 1px 10px;       /* Innerer Abstand: „oben,unten 1 pixel, links,rechts 10 pixel“ */
}
</style>
```

4.3 Slider mit mehreren Bereichen

Im Gegensatz zum ersten Beispiel, welches ja mehr oder weniger nur ein bisschen „Leben in die Bude“ bringen sollte, ist dieses zweite Beispiel eine Erweiterung der Slider-Funktionalität.

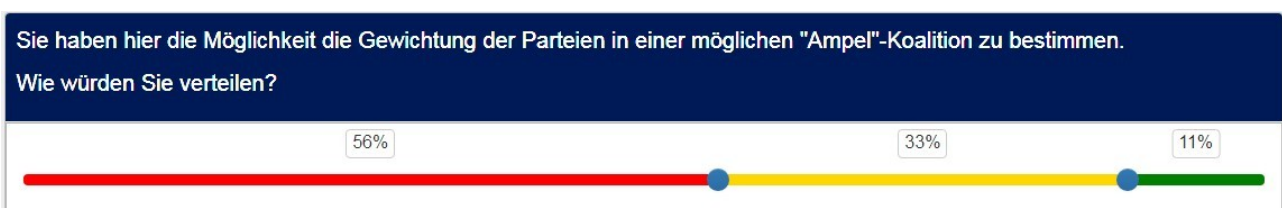
4.3.1 Anwendungsbeispiel

Und zwar können hier 100% auf drei Bereiche verteilt werden.

Klar, dazu kann man auch eine Frage vom Typ „mehrfache Zahleingabe“ nehmen (auf einer solchen beruht dieses Beispiel auch).

Insofern ist es eigentlich auch nur „Gamification“.

Nehmen wir ein aktuelles Beispiel:



4.3.2 Implementierung

Vorweg:

Wieder viel javascript und viel css.

Das alles wiederum von Tony Partner (tpartner).

Es gibt dies auch als Fragenvorlage hier

Version 3.x. : <https://github.com/tpartner/LimeSurvey-Range-Slider-3x>

Version 4.x/5.x : <https://github.com/tpartner/LimeSurvey-Range-Slider-4x-5x>

und auch im LimeStore hier: <https://account.limesurvey.org/de/limestone>

Aber jetzt zur javascript-Lösung.

Dies alles – wie immer – in den Fragetext (im Quellcode-Modus)

javascript:

```
<script type="text/javascript" charset="utf-8">
    $(document).on('ready pjax:scriptcomplete',function(){

        // Identify this question
        var qID = '{QID}';
        var thisQuestion = $('#question'+qID);

        // Path to the LS installation root
        var rootPath = location.pathname.split('index.php')[0];

        // Hide the answer inputs
        $('#subquestion-list', thisQuestion).hide();

        // Insert an input for the slider
        $('#answer-container', thisQuestion).prepend('<div class="inserted-slider-wrapper"><input
id="range'+qID+'" type="text" value="" /></div>');

        // Include the Bootstrap slider styles
        $('head').append('<link rel="stylesheet"
href="'+rootPath+'assets/packages/bootstrap/plugins/slider/css/bootstrap-slider.css"
type="text/css" />');

        // Get the Bootstrap slider script
        $.getScript(rootPath+'assets/packages/bootstrap/plugins/slider/bootstrap-slider.js')
            .done(function( script, textStatus ) {

                //Initial slider values
                var lowValue = 33;
                var highValue = 67;
                var emptyInputs = $(':text.form-control', thisQuestion).filter(function() {
                    return $.trim($(this).val()) == '';
                });
                if(emptyInputs.length == 0) {
                    lowValue = Number($(':text.form-control:eq(0)', thisQuestion).val());
                    highValue = Number($(':text.form-control:eq(0)', thisQuestion).val()) + Number($
(':text.form-control:eq(1)', thisQuestion).val());
                }

                // Initate the range slider
                $('#range'+qID+'.').slider({
                    min: 0,
                    max: 100,
                    step: 1,
                    range: true,
                    value: [lowValue,highValue]
                })
            })
    })
</script>
```

```

        // Listener on the slider
        .on('change', function(e) {
            var lowPercent = round((e.value.newValue[0]/100)*100);
            var highPercent = round(100 - (e.value.newValue[1]/100)*100);
            var middlePercent = round(100 - (highPercent+lowPercent));

            // Load the answer inputs
            $(':text.form-control:eq(0)', thisQuestion).val(lowPercent).trigger('keyup');
            $(':text.form-control:eq(1)', thisQuestion).val(middlePercent).trigger('keyup');
            $(':text.form-control:eq(2)', thisQuestion).val(highPercent).trigger('keyup');

            handleTooltips();
        });

        // Insert some custom tooltips
        $('.slider-track > div', thisQuestion).append('<div class="inserted-tooltip
hidden" />');
        handleTooltips();
    })
    .fail(function( jqxhr, settings, exception ) {
        console.log( exception );
    });

    // A function to handle the custom tooltips
    function handleTooltips() {
        $(':text.form-control', thisQuestion).each(function(i) {
            if($(this).val() != '') {
                $('.inserted-tooltip:eq('+i+')', thisQuestion).text($(this).val()
+'%').removeClass('hidden');
            }
        });
    }
});
</script>

```

Und css:

```

<style type="text/css">.inserted-slider-wrapper .slider.slider-horizontal {
    width: 100%;
    margin: 40px 0 20px 0;
}

.inserted-slider-wrapper .slider-track,
.inserted-slider-wrapper .slider-selection {
    box-shadow: none;
    background: transparent none;
}

.inserted-slider-wrapper .slider-track-low {
    background: red;
}

```

```
.inserted-slider-wrapper .slider-selection {
  background: gold;
}

.inserted-slider-wrapper .slider-track-high {
  background: green;
}

.inserted-slider-wrapper .inserted-tooltip {
  position: absolute;
  top: -40px;
  left: 50%;
  padding: 0 5px 2px 5px;
  border: 1px solid #ccc;
  border-radius: 4px;
  transform: translateX(-50%);
}

.inserted-slider-wrapper .tooltip {
  display: none !important;
}

</style>
```

5 Drop-Down

5.1 Drop-Down als Option in Einfach- und Mehrfachnennung

5.1.1 Anwendungsbeispiel

Manchmal ist es sicherlich angebracht, auch eine eigentlich „Sonstige“-Nennung zu strukturieren.

The image shows a survey interface for question Q1... The instruction is 'Bitte wählen Sie eine der folgenden Antworten:'. There are six radio button options: 'Adalbert Ameise', 'Berta Biber', 'Carlo Chinchilla', 'Doris Dachs', 'Anderer Kandidat' (which is selected), and 'Andere Kandidatin'. A dropdown menu is open for the 'Anderer Kandidat' option, showing a list of names: 'Erich Eidechse', 'Bitte auswählen..', 'Erich Eidechse' (highlighted), 'Florian Faultier', 'Gustav Guanaco', and 'Harald Haubentaucher'.

Die Idee dahinter ist klar. Optionen, bei denen man davon ausgeht, dass sie nicht besonders häufig gewählt werden, werden hierhin ausgelagert.

Damit erspart man sich, dass die Liste zu lang wird, man erspart sich „Lesen und Umkodieren“ bei einer freien Texteingabe, hier könnte eben der Kandidat mit den verschiedensten Schreibweisen auftauchen.

5.1.2 Implementierung

5.1.2.1. Einfachnennung

Es werden nach der „Liste(Optionsfelder)“-Frage Fragen vom Typ „Liste(Klappbox)“ erstellt.

Eben so viele wie es in der Liste(Optionsfelder) diese Drop-Downs geben soll. (Hier im Beispiel sind es zwei)

Die erste Liste enthält auch die Punkte „Andere...“.

Mit dem folgenden javascript werden nun die Drop-Downs aus den folgenden Fragen in diese hineingezogen.

Wie immer wird also dieses script (im Quellcode-Modus) in den Fragetext eingefügt.

```
<script type="text/javascript" data-author="Tony Partner">

$(document).on('ready pjax:scriptcomplete',function(){
    // The text for the "Please choose" option
    var chooseText = {
        'en': 'Please choose...',
        'de': 'Bitte auswählen...',
        'fr': ' Veuillez choisir ...'
    }
    // Identify the questions
    var qID = '{QID}';
    var thisQuestion = $('#question'+qID);
    var nextQuestion = $(thisQuestion).nextAll('.list-dropdown:eq(0)');
    var nextQuestion2 = $(thisQuestion).nextAll('.list-dropdown:eq(1)');

    var lang = $('html').attr('lang');

    // Hide the next questions
    nextQuestion.hide();
    nextQuestion2.hide();

    // Move the dropdowns
    $('.answer-item.radio-item:eq(4)', thisQuestion).addClass('with-dropdown').append($
    ('.answer-item', nextQuestion));
    $('.answer-item.radio-item:eq(5)', thisQuestion).addClass('with-dropdown').append($
    ('.answer-item', nextQuestion2));

    // Cleanup styles
    $('.answer-item.radio-item .answer-item', thisQuestion).css({
        'display': 'inline-block',
        'margin-left': '1em',
        'padding': '0'
    });

    // Initial states
    var iChooseText = chooseText['en'];
    if(lang in chooseText) {
        iChooseText = chooseText[lang];
    }
});
```



```

    }
    $.each($('select.form-control', thisQuestion), function(i) {
        if($('option[value=""]', this).length == 0) {
            $(this).prepend('<option value="">' + iChooseText + '</option>');
        }
    });

    // Listener on the radios
    $('answer-item.radio-item :radio', thisQuestion).on('click', function(e) {
        $.each($('answer-item.with-dropdown', thisQuestion), function(i) {
            if($(':radio:checked', this).length == 0) {
                $('answer-item input[type="hidden"]', this).val('');
                $('answer-item select', this).val('').trigger('change');
            }
        });
    });

    // Listener on the dropdowns
    $('answer-item.with-dropdown select', thisQuestion).on('change', function(e) {
        if($(this).val() != '') {
            $(this).closest('radio-item').find(':radio').trigger('click');
        }
    });
});
</script>

```

Zu beachten wäre noch dieses:

- In diesen Zeilen
`var nextQuestion = $(thisQuestion).nextAll('.list-dropdown:eq(0)');`
`var nextQuestion2 = $(thisQuestion).nextAll('.list-dropdown:eq(1)');`
`var nextQuestion3 = $(thisQuestion).nextAll('.list-dropdown:eq(2)');`
 werden Variable gebildet, welche die folgenden Fragen beinhalten.
- In der Variablen „chooseText“ ganz am Anfang können für mehrsprachige Fragebögen die Aufforderungstexte der Drop-Downs gewählt werden.
- In den Zeilen
`$('.answer-item.radio-item:eq(4)', thisQuestion).addClass('with-dropdown').append($`
`('answer-item', nextQuestion));`
`$('.answer-item.radio-item:eq(5)', thisQuestion).addClass('with-dropdown').append($`
`('answer-item', nextQuestion2));`
 wird in „:eq(x)“ die Option ausgewählt, bei welcher das Drop-Down erscheinen soll. Start der Zählung ist „0“. Und hier werden dann die oben erwähnten Variablen zum Einfügen benutzt.

5.1.2.2. Mehrfachnennungen

Hier wird analog gearbeitet und folgendes script benutzt.

```
<script type="text/javascript" data-author="Tony Partner">

$(document).on('ready pjax:scriptcomplete',function(){
    // The text for the "Please choose" option
    var chooseText = {
        'en': 'Please choose...',
        'de': 'Bitte auswählen...',
        'fr': ' Veuillez choisir ...'
    }
    // Identify the questions
    var qID = '{QID}';
    var thisQuestion = $('#question'+qID);
    var nextQuestion = $(thisQuestion).nextAll('.list-dropdown:eq(0)');
    var nextQuestion2 = $(thisQuestion).nextAll('.list-dropdown:eq(1)');

    var lang = $('html').attr('lang');

    // Hide the next questions
    nextQuestion.hide();
    nextQuestion2.hide();

    // Move the dropdowns
    $('.answer-item.checkbox-item:eq(0)', thisQuestion).addClass('with-dropdown').append($
    ('.answer-item', nextQuestion));
    $('.answer-item.checkbox-item:eq(1)', thisQuestion).addClass('with-dropdown').append($
    ('.answer-item', nextQuestion2));

    // Cleanup styles
    $('.answer-item.checkbox-item .answer-item', thisQuestion).css({
        'display': 'inline-block',
        'margin-left': '1em',
        'padding': '0'
    });

    // Initial states
    var iChooseText = chooseText['en'];
    if(lang in chooseText) {
        iChooseText = chooseText[lang];
    }
    $.each($('.select.form-control', thisQuestion), function(i) {
        if($('.option[value=""]', this).length == 0) {
            $(this).prepend('<option value="">'+iChooseText+'</option>');
        }
    });

    // Listener on the checkboxes
    $('.answer-item.checkbox-item :checkbox', thisQuestion).on('change', function(e) {
        $.each($('.answer-item.with-dropdown', thisQuestion), function(i) {
```

```
        if($(':checkbox:checked', this).length == 0) {
            $('answer-item input[type="hidden"]', this).val('');
            $('answer-item select', this).val('').trigger('change');
        }
    });
});

// Listener on the dropdowns
$('answer-item.with-dropdown select', thisQuestion).on('change', function(e) {
    var thisItem = $(this).closest('checkbox-item');
    if($(this).val() != '') {
        $('input[type="hidden"]:first', thisItem).val('Y');
        $(':checkbox', thisItem).prop('checked', true).trigger('change');
    }
    else {
        $('input[type="hidden"]:first', thisItem).val('');
        $(':checkbox', thisItem).prop('checked', false).trigger('change');
    }
});
});
</script>
```

6 Anhang

6.1 Iss-Export einer Beispiel-Datei

Hier ist der Link zu einer gezippten Iss-Datei, die die hier beschriebenen Beispiele enthält.

https://www.Mafosurvey.de/lime/Tutorial_Mehrfach.zip

6.2 Zusätzliche benötigte Bibliotheken und Dateien

6.2.1 Beispiel-Text-Dateien für „autocomplete“

https://www.Mafosurvey.de/lime/Beispiel_Text_Dateien.zip

6.2.2 Zusätzliche Bibliotheken für „autocomplete“

https://www.Mafosurvey.de/lime/jquery_Bibliotheken.zip