aim werbeagentur · Müllenhoffstraße 1 · D-10967 Berlin

aim werbeagentur gmbh

Müllenhoffstraße 1
10967 Berlin

Tel. 030 / 69 81 63 70

f.zeitz@aim-schulungen.de
www.aim-schulungen.de

Zeitberechnung in Adobe Acrobat

Berlin am 31. Dezember 2017,
Kleines Skript für die Berechnung von Uhrzeit-Differenzen und deren Darstellung auch in Dezimalzeit,
aufgeschrieben von Frank Zeitz

Von	Bis	Stunden u. Minuten	Dezimalzeit
01:23	04:56	3:33	3,55

Der Code

```
1  var dtStart = this.getField("von").value;
2  var dtEnde  = this.getField("bis").value;
3  dtStart = dtStart.toString();
4  dtStart = dtStart.split(":");
5  var dtStartH = Number(dtStart[0]);
6  var dtStartM = Number(dtStart[1]);

7  dtEnde = dtEnde.toString();
8  var dtEnde = dtEnde.split(":");
9  var dtEndeH = Number(dtEnde[0]);
10 var dtEndeM = Number(dtEnde[1]);

11 var zeitStart = dtStartH * 60 + dtStartM;
12 var zeitEnde  = dtEndeH * 60 + dtEndeM;

13 var zeitDiff = zeitEnde - zeitStart;

14 var minuten = zeitDiff % 60;
15 var stunden = (zeitDiff-minuten) / 60;

16 var minutenausdruck = minuten;

17 if (minuten < 10) {
18     minutenausdruck = "0"+minuten;
19 }

20 this.getField("dauer").value = stunden + ":" + minutenausdruck;

21 var dezimalzeit = Number(stunden) + Number(minuten/60);
22 dezimalzeit = dezimalzeit.toFixed(3);

23 this.getField("dezimaldauer").value = dezimalzeit;
```

Dieses Dokument ist interaktiv. Probieren Sie die Funktionalität einfach oben in den vier Feldern aus!

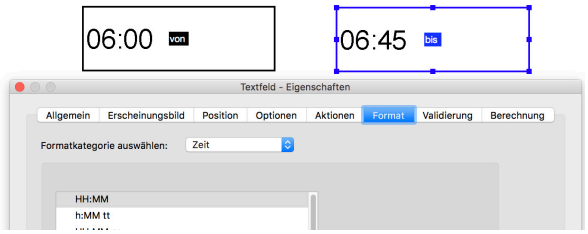
Erläuterung des Codes

Das Rechnen mit Zeit ist in Excel ja sehr einfach, wir hatten da einen kurzen Exkurs im Seminar.

In JavaScript ist es eigentlich auch nicht viel anders, nur evtl. etwas ungewohnt. Ich habe das Skript, das ich dafür geschrieben habe, deshalb sehr ausführlich, und damit hoffentlich auch nachvollziehbar gehalten.

Grundgedanke

Ich habe zwei Felder „*von*“ und „*bis*“, die ich so formatiere, daß nur Uhrzeiten eingegeben werden können.



Ich lese diese im Skript im zwei Variablen ein, und konvertiere die Variable dann in einen normalen „String“, also eine Zeichenkette [*vgl. Seite 1, Zeilen 1 bis 3 und 7*]:

```
var dtStart = this.getField("von").value;
var dtEnde  = this.getField("bis").value;
dtStart = dtStart.toString();
```

Dann splitte ich beide Werte jeweils in einen Stunden und einen Minutenwert (die Trennmarke ist der Doppelpunkt der Uhrzeit) [*vgl. Seite 1, Zeilen 4 bis 6 und 8 bis 10*]:

```
dtStart = dtStart.split(":");
var dtStartH = Number(dtStart[0]);
var dtStartM = Number(dtStart[1]);
var dtEnde = dtEnde.split(":");
var dtEndeH = Number(dtEnde[0]);
var dtEndeM = Number(dtEnde[1]);
```

Diese vier Werte wandle ich dann in Zahlen um [*vgl. Seite 1, Zeilen 11 bis 12*]:

```
var zeitStart = dtStartH * 60 + dtStartM;
var zeitEnde  = dtEndeH * 60 + dtEndeM;
```

So bekomme ich Gesamtminuten, die ich voneinander abziehen kann [*vgl. Seite 1, Zeile 13*]:

```
var zeitDiff = zeitEnde - zeitStart;
```

Diese rechne ich dann wieder in Stunden um. Indem ich die Zeitdifferenz *zeitDiff* modulo *60* rechne, erhalte ich die Restminuten. Das ist die Zahl, die ich rechts vom Doppelpunkt anzeigehn möchte.

Die Restminuten ziehe ich dann von *zeitDiff* ab und teile diesen Wert durch *60*. Dadurch erhalte ich genau die Stunden,die ich links des Doppelpunkts anzeigen möchte [*vgl. Seite 1, Zeilen 14 bis 15*]:

```
var minuten = zeitDiff % 60;
var stunden = (zeitDiff-minuten) / 60;
```

Das ist alles, das benötigt wird für die Anzeige im dritten Feld, also im Feld „Stunden und Minuten“. Falls ich weniger als 10 Restminuten habe, benötige ich noch eine *0*, die ich diesen voranstelle, damit ich eine schöne Minutenanzeige erhalte. Das ist aber auch schon alles [*vgl. Seite 1, Zeilen 16 bis 20*]:

```
var minutenausdruck = minuten;
if (minuten < 10) {
    minutenausdruck = "0"+ minuten;
}
this.getField("dauer").value = stunden + ":" + minutenausdruck;
```

Da man aber mit diesem Wert nicht rechnen kann, also jedenfalls keine sinnvolle Multiplikation mit Eurobeträgen, habe ich noch ein viertes Feld eingefügt, das den Zeitwert in sogenannte Dezimalzeit umwandelt.

Sowohl die stunden also auch die minuten Variable sind ja Strings. Deshalb müssen diese erst mithilfe des Befehls Number() in Zahlen zurückverwandelt werden. Der Wert für Minuten muß dabei natürlich durch 60 geteilt werden. Jetzt entsteht gegebenenfalls eine lange Gleitkommazahl. Ich habe deshalb die Anzahl der Nachkommestellen auf drei reduziert. Und anschließend den Wert in das vierte Feld „Dezimalzeit“ übertragen [*vgl. Seite 1, Zeilen 20 bis 23*]:

```
var dezimalzeit = Number(stunden) + Number(minuten/60);
dezimalzeit = dezimalzeit.toFixed(3);
this.getField(„dezimaldauer“).value = dezimalzeit;
```

Die Dezimalzeit kann dann für weitere Berechnungen, z.B. für Multiplikation mit einem Stundensatz, verwendet werden.

Sicherlich kann man das Skript sehr viel kürzer fassen, und damit sicher auch etwas effizienter, aber mein Ziel war hier, wirklich jeden Einzelschritt, der gedanklich notwendig ist, in eine eigene Programmierzeile zu packen.
Dieses kleine Skript ist auch überhaupt nicht vollständig. Wenn man zum Beispiel für die „Bis“ Zeit einen niedrigeren Wert eingibt als für die „Von“ Zeit, dann ist das Ergebnis, nun ja, mindestens etwas unbefriedigend.

Hier könnte man sicherlich sehr leicht mithilfe einer Wenn-Dann-Bedingung [*vgl. die Zeilen 17-19 auf Seite 1*] dafür sorgen, daß immer ein positiver Wert für *zeitDiff* berechnet wird.

Ich hoffe, Sie können das gut nachvollziehen.
Falls an einer Stelle nicht: Einfach nachfragen!