

Diese Funktion überprüft einen eingegebenen Text nach folgenden Regeln:

- Nur Zahlen, Punkte und ein Komma sind erlaubt
- Falls ein Komma vorhanden ist, müssen zwei Nachkommastellen folgen
- Eine führende Null darf nur angegeben werden, wenn direkt danach ein Komma folgt

Werden diese Regeln durch die Eingabe verletzt, erscheint ein Popup mit dem jeweiligen Hinweistext. Die Eingabe wird nichtsdestotrotz im Feld angezeigt und nach dem Speichern der Smartform auch in die Datenbank geschrieben.

Außerdem nimmt die Funktion folgende Formatierung vor:

- Falsch gesetzte Tausendertrennzeichen („.“) werden gelöscht
- Die korrekten Tausendertrennzeichen werden eingefügt
- Falls kein Komma vorhanden ist, wird an die Eingabe „,00“ angefügt
- Beispiel: die Eingabe „1234567“ wird formatiert zu „1.234.567,00“

Die Funktion muss als onChange-Funktion des jeweiligen Input-Elementes gesetzt werden.

Geldbeträge überprüfen

```
gadget.functions.checkMonetaryValue = function() {  
  
    var isANumber = function(stringGot) {  
        if(stringGot.length == 0)  
            return false;  
        for(var i=0;i<stringGot.length;i++) {  
            if(stringGot.charAt(i) != "0" && stringGot.charAt(i) != "1"  
&& stringGot.charAt(i) != "2" && stringGot.charAt(i) != "3" &&  
stringGot.charAt(i) != "4" &&  
                stringGot.charAt(i) != "5" && stringGot.charAt(i)  
!= "6" && stringGot.charAt(i) != "7" && stringGot.charAt(i) != "8" &&  
stringGot.charAt(i) != "9") {  
                return false;  
            }  
        }  
        return true;  
    }  
  
    var wert = this.value;  
    if(this.value == "" || this.value == " ") {  
        this.entity.setValue(this.id,this.value);  
        return;  
    }  
    var splitNachKommas = this.value.split(",");  
    if(splitNachKommas.length > 2) { //check, ob mehr als ein komma  
vorhanden  
        alert("Bitte geben Sie nur ein Komma ein!");  
        this.focus();  
        this.entity.setValue(this.id,this.value);  
        return;  
    }  
}
```

```
if(splitNachKommas.length == 2) { //ein Komma vorhanden
    if(splitNachKommas[1].length != 2) {
        alert("Bitte geben Sie zwei Nachkommastellen an!");
        this.focus();
        this.entity.setValue(this.id,this.value);
        return;
    }
    if(splitNachKommas[1].length == 2) { //zwei Nachkommastellen
    vorhanden
        if(!isANumber(splitNachKommas[1])) {
            alert("Die Nachkommastellen dürfen nur Ziffern
            enthalten!");
            this.focus();
            this.entity.setValue(this.id,this.value);
            return;
        }
    }
}

var splitNachPunkten = this.value.split(".");
if(splitNachPunkten.length > 1) { //mind. ein punkt vorhanden

    var neuerValue = splitNachKommas[0].replace(/\.\/g,"")
    var ende = neuerValue.length;
    var counter = 0;
    var somethingChanged = false;
    for(var j=ende-1;j>=0;j--) {
        counter++;
        if(counter == 3 && j != 0) {
            neuerValue = neuerValue.substring(0,j) + "." +
            neuerValue.substr(j);
            counter = 0;
            //somethingChanged = true;
        }
    }
    //if(somethingChanged) {
        if(splitNachKommas.length == 2) {
            neuerValue = neuerValue + "," + splitNachKommas[1];
        }
        this.value = neuerValue;
        splitNachPunkten = this.value.split(".");
        splitNachKommas = this.value.split(",");
        this.entity.setValue(this.id,this.value);
    //}

    for(var i=0; i<splitNachPunkten.length; i++) {
        /* MOE 17.07.13: NICHT MEHR BENÖTIGT; DA JETZT DIE
```

```

TRENnzeichen AUTOMATISCH GESETZT WERDEN!
    if(splitNachPunkten[i].split(",")[0].length != 3 && i != 0)
{ //nach einem punkt folgen nicht 3 zeichen und es ist nicht der teil
vorn ersten punkt
    alert("Bitte überprüfen Sie die
Tausendertrennzeichen!");
    this.focus();
    this.entity.setValue(this.id,this.value);
    return;
}*/
    if(!isANumber(splitNachPunkten[i].split(",")[0])) { //die 3
zeichen sind entweder keine zahl oder es befindet sich ein leerzeichen
darunter
        alert(wert + " ist kein zulässiger Zahlenwert. Bitte
geben Sie eine gültige Zahl ein.");
        this.focus();
        this.entity.setValue(this.id,this.value);
        return;
    }
}

for(g=0;g<splitNachKommas[0].length;g++) {
    if(splitNachKommas[0].charAt(g) == "0" && g == 0) { //
Leerzeichen in der Zahl vorhanden
        if(splitNachKommas[0].length > 1) {
            alert("Bitte entfernen Sie die führenden Nullen!");
            this.focus();
            this.entity.setValue(this.id,this.value);
            return;
        }
    }
}

}
else { //kein Punkt vorhanden
    if(!isANumber(splitNachKommas[0])) { //Zahl hat nicht nur
Ziffern
        alert(wert + " ist kein zulässiger Zahlenwert. Bitte geben
Sie eine gültige Zahl ein.");
        this.focus();
        this.entity.setValue(this.id,this.value);
        return;
    }
    for(g=0;g<splitNachKommas[0].length;g++) {
        if(splitNachKommas[0].charAt(g) == "0" && g == 0) { //
Leerzeichen in der Zahl vorhanden
            if(splitNachKommas[0].length > 1) {
                alert("Bitte entfernen Sie die führenden Nullen!");
                this.focus();
            }
        }
    }
}
}

```

```
        this.entity.setValue(this.id,this.value);
        return;
    }
}

}

var ende = splitNachKommas[0].length;
var counter = 0;
var neuerValue = splitNachKommas[0];
var somethingChanged = false;
for(var j=ende-1;j>=0;j--) {
    counter++;
    if(counter == 3 && j != 0) {
        neuerValue = neuerValue.substring(0,j) + "." +
neuerValue.substr(j);
        counter = 0;
        somethingChanged = true;
    }
}
if(somethingChanged) {
    if(splitNachKommas.length == 2) {
        neuerValue = neuerValue + "," + splitNachKommas[1];
    }
    this.value = neuerValue;
    this.entity.setValue(this.id,this.value);
}
}

if(splitNachKommas.length == 1) { //kein Komma vorhanden
    this.value = this.value + ",00";
    this.entity.setValue(this.id,this.value);
}
this.entity.setValue(this.id,this.value);
//focus muss im feld bleiben, falls etwas nicht passt
}
```

From: <https://wiki.tim-solutions.de/> - [TIM Wiki](#) / [NEW TIM 6 Documentation](#)

Permanent link: https://wiki.tim-solutions.de/doku.php?id=projekt:jsfundgrube:check_monetary_value&rev=1374156920

Last update: **2021/07/01 09:53**

