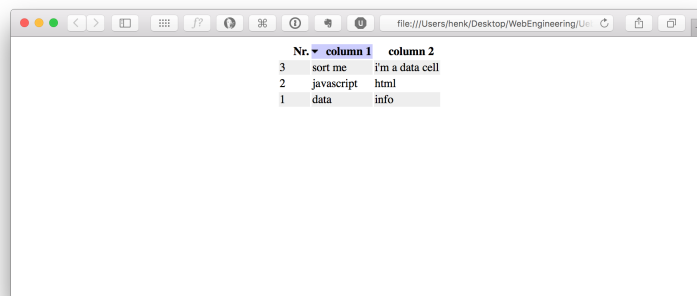


0.1 Übung 7 - jQuery (3)

Aufgabe (a)

Implementieren Sie mithilfe von jQuery eine sortierbare Tabellenansicht. Für jede Spalte der Tabelle soll durch Klicken eine Sortierung in auf- und absteigender Reihenfolge wählbar sein. Die Tabelle soll sowohl Spalten mit numerischem als auch Spalten mit alphanumerischem Inhalt darstellen. Machen Sie Verwendung von JavaScript `sort()` und definieren Sie eine Komparator-Funktion, die für beide Datentypen die richtige Sortierung ermöglicht. Verwenden Sie die vorgegebenen JavaScript-Funktionen sowie CSS-Definitionen aus den dieser Übung beigelegten Vorlagen.



Nr.	column 1	column 2
3	sort me	i'm a data cell
2	javascript	html
1	data	info

Abbildung 0.1: Eine sortierte Tabelle

Die Funktion `sort()` kann auf beliebige Arrays angewandt werden. Dem Sortierprozess liegt eine Vergleichsfunktion für die einzelnen Array-Elemente zugrunde, die standardmäßig nur die Unicode-Werte einzelner Zeichen vergleicht. Die folgenden Beispiele zeigen zwei exemplarische Sortierungen mit dem default-Komparator.

```
1 var fruit = ['cherries', 'apples', 'bananas'];
2 fruit.sort();
3 // ['apples', 'bananas', 'cherries']
4
5 var numbers = [1, 10, 2, 21];
6 numbers.sort();
7 // [1, 10, 2, 21]
```

Damit auch numerische Werte korrekt sortiert werden, muss beim Aufruf der Sortierfunktion eine geeignete Komparator-Funktion übergeben werden. Im einfachsten Fall lässt sich so ein Komparator wie folgt realisieren:

```
1 var numbers = [4, 2, 5, 1, 3];
2 numbers.sort(function(a, b) {
3     return a - b;
```

```

4  });
5  console.log(numbers);
6
7  // [1, 2, 3, 4, 5]

```

Im Falle der alphabetischen Sortierung sollten die zu sortierenden Strings zunächst in die Kleinschreibweise überführen. Hierzu empfiehlt sich `string.toLowerCase()`.

Aufgabe (b)

Beim Sortieren von Spalten mit Umlauten und anderen besonderen Buchstaben (ä, ö, ü, ß) fällt auf, dass diese hinter dem regulären Alphabet eingeordnet werden, was durch ihre Wertigkeit in der Unicode-Tabelle bedingt ist. Implementieren Sie die Funktion `replaceSpecials`, die diese Zeichen in der Komparator-Funktion zu ihren Verwandten einordnet (ä zu a, ö zu o usw.).

Mit JavaScript `RegExp` und `string.replace()` lassen sich Umlaute in Strings durch ihre alternativen Schreibweisen ersetzen, siehe folgendes Beispiel:

```

1  var str = "Hällo World";
2  var regExp = new RegExp('ä', 'g');
3  str = str.replace(regExp, 'e'); // Hello World

```

Der Parameter `'g'` beim Erstellen von `regExp` ermöglicht ein globales Matching, andernfalls würde das Matching nach dem ersten Treffer beendet werden.