Multimedia im Netz

Wintersemester 2013/14

Übung 06 (Nebenfach)

JavaScript: Events (I)

```
<!DOCTYPE html>
<html lang="de">
<head><title>HTML5</title></head>
<body>
<!-- Variante 1 -->
<input type="button" name="text" value="Click 1" onclick="callMe()"/>
<!-- Variante 2 -->
<input type="button" name="text" value="Click 2" onclick="callMe2(this)"/>

<script>
    //Variante 1
    function callMe(){
        alert("Hi");
    }

    //Variante 2
    function callMe2(element){
        alert("Button Value: " + element.value);
    }
</script>
</body></html>
```

JavaScript: addEventListener

```
<!DOCTYPE html>
<html lang="de">
<head>
<title>HTML5</title>
</head>
<body>
<!-- Variante 3 -->
<input id="button3" type="button" name="text" value="Click
3"/>

<script>
//Variante 3
var button3 = document.getElementById("button3");
button3.addEventListener("click", function(){
                    alert("Button Value: " + this.value);});
</script>
</body></html>
```

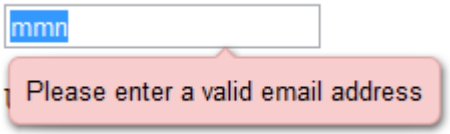
Event Types

- Maus Events
  - onclick
  - ondbclick
  - onmousedown
  - onmouseover
  - …
- Tastatur Events
  - onkeydown
  - onkeyup
  - …
- …

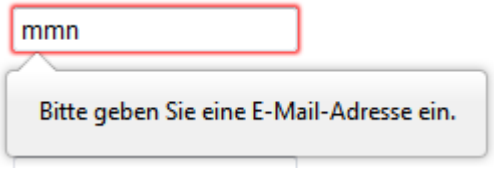
HTML5 Form Validation

- Zusätzliches semantisches Markup für Formularelmente in HTML5 ermöglichen es zum Beispiel Nutzereingaben vor dem Absenden zu überprüfen.

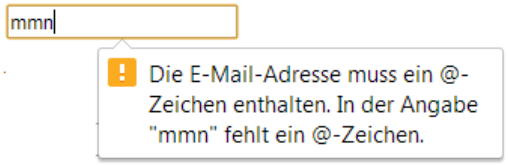
- Opera



- Firefox



- Chrome



- Safari

HTML5: Neue Input Types

- **E-Mail**

```
<input type="email" name="email" />
```

- **URL**

```
<input type="url" name="url"/>
```

- **…**

HTML5: Neue Attribute

- **Formular nicht validieren (geht auch für input-Elemente)**
```
<form novalidate>
```

- **Platzhalter**
```
<input type="email" name="email" placeholder="Email" />
```

- **Required**
```
<input type="email" name="email" required />
```

- **Autofokus**
```
<input type="email" name="email" autofocus />
```

- **Pattern**
```
<input type="text" name="text" pattern="[a-zA-Z]"/>
```

Pattern / Reguläre Ausdrücke

- Mit einem regulären Ausdruck kann vorgegeben werden, wie eine bestimmte Zeichenkette aussehen muss.

- In dem vorigen Beispiel:

```
<input type="text" name="text" pattern="[a-zA-Z]"/>
```

  ➔ Man darf in das Textfeld nur Groß- und Kleinbuchstaben eingeben

Reguläre Ausdrücke: Zeichenauswahl

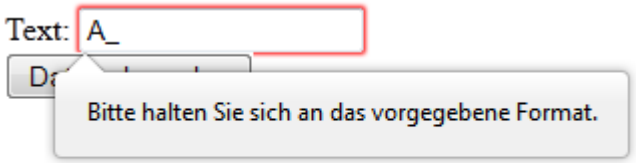
[0-6]	Eine der Ziffern von 0 bis 6
[A-Za-z]	Eines der Zeichen von A-Z oder a-z
[A-Za-z0-9]	Eines der Zeichen von A-Z oder a-z oder eine der Ziffern von 0-9
[^z]	Ein beliebiges Zeichen außer a
\d	Eine Ziffer von 0-9
\D	Ein Zeichen, das keine Ziffer ist
\w	Ein Buchstabe, eine Ziffer oder ein Unterstrich
\W	Ein Zeichen, das nicht \w ist
\s	Leeraum
\S	Ein Zeichen, das nicht \s ist

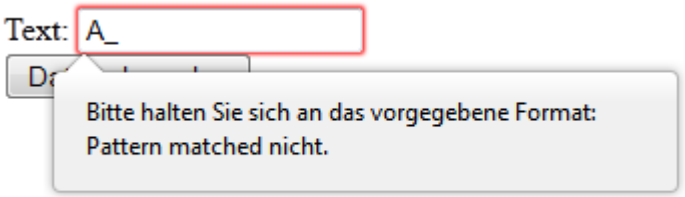
Reguläre Ausdrücke: Quantoren

- Wie oft darf ein Zeichen vorkommen?

?	Vorhergehender Ausdruck ist optional: kann einmal oder keinmal vorkommen
+	Vorhergehender Ausdruck muss mindestens einmal vorkommen
*	Vorhergehender Ausdruck kann beliebig oft vorkommen
{n}	Vorhergehender Ausdruck darf genau n-mal vorkommen
{min, }	Vorhergender Ausdruck muss mindestens min-mal vorkommen
{min, max}	Vorhergehender Ausdruck muss mindestens min-mal und kann maximal max-mal vorkommen

Fehlermeldungen anpassen (I)






```
…
<form>
    <label for="text">Text: </label>
    <input id="text" type="text"
           name="text"
           pattern="[a-zA-Z0-9]+"
           title="Pattern matched nicht."
           required />
    <input type="submit" id="submit" />
</form>
…
```

Constraint Validation API

- Bietet weitere Eigenschaften / Methoden um mit DOM Knoten zu arbeiten
- **validity:** Gibt ein Objekt ValidityState zurück
- **ValidityState** hat verschiedene Attribute, die true oder false sein können. Zum Beispiel:
  - **valueMissing**
  - **typeMismatch**
  - **patternMismatch**
- **checkValidity()** überprüft, ob alle ValidityStates eines Elements „true“ sind
- **setCustomValidity()** setzt die angepasste Fehlermeldung

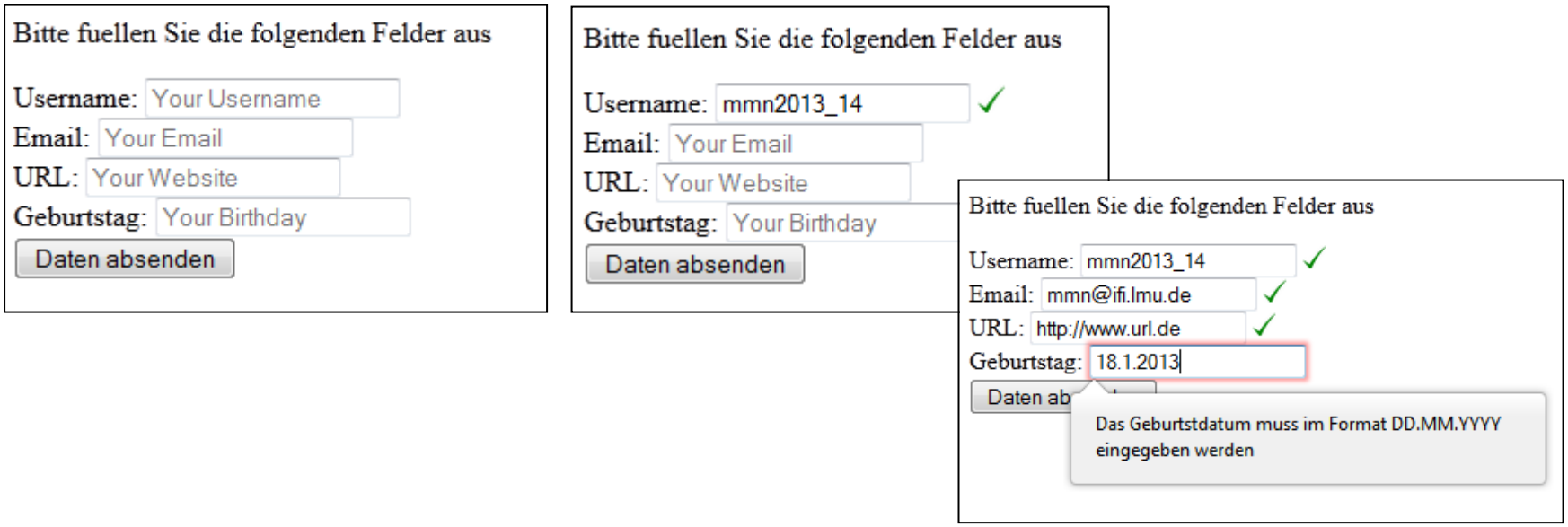
Fehlermeldungen anpassen (II)

```
<!DOCTYPE html>
<html lang="de">
<head><title>HTML5</title></head>
<body>
<form>
<label for="text">Text: </label>
<input id="text" type="text" name="text" pattern="[a-zA-Z0-9]+" required />
<input type="submit" id="submit" />
</form>

<script>
var text = document.getElementById("text"); text.addEventListener("keyup",
function(){
    if(this.validity.patternMismatch){
        this.setCustomValidity("Pattern matched nicht.");
    } else {
        this.setCustomValidity("");
    }
});
</script>
</body>
</html>
```

Übungsblatt 6

- **Thema: JavaScript und HTML5 Formulare**
- Bearbeitungszeit: 1 Woche
- Abgabe: 11.12.2013 23:00 Uhr



Danke! Fragen?