Kostenloses eBook

LERNEN HTML

Free unaffiliated eBook created from
Stack Overflow contributors.

#html

Inhaltsverzeichnis

Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: html

It is an unofficial and free HTML ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official HTML.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Kapitel 1: Erste Schritte mit HTML

Bemerkungen

HTML **(H** yper **t** ext **M** arkup **L** anguage) ist ein XML - konformes System von Dokumenten mit 'Tags' mit Anmerkungen versehen. Es wird speziell zum Erstellen von Inhalten für Webseiten und Webanwendungen verwendet, die dann über ein Netzwerk gemeinsam genutzt werden können.

Neben Text unterstützt die aktuelle Version von HTML viele verschiedene Medientypen , einschließlich Bilder und Videos.

Versionen

Ausführung	Spezifikation	Veröffentlichungsdatum
1,0	N / A	1994-01-01
2,0	RFC 1866	1995-11-24
3.2	W3C: HTML 3.2-Spezifikation	1997-01-14
4,0	W3C: HTML 4.0-Spezifikation	1998-04-24
4.01	W3C: HTML 4.01-Spezifikation	1999-12-24
5	WHATWG: HTML Living Standard	2014-10-28
5.1	W3C: HTML 5.1-Spezifikation	2016-11-01

Examples

Hallo Welt

Einführung

HTML (**H** yper **t** ext **M** arkup **L-** Sprache) verwendet ein Auszeichnungssystem, das aus Elementen besteht, die bestimmten Inhalt darstellen. *Markup* bedeutet, dass Sie mit HTML deklarieren, *was* einem Betrachter präsentiert wird, und nicht *wie* es dargestellt wird. Visuelle Repräsentationen werden von Cascading Style Sheets (CSS) definiert und von Browsern umgesetzt. Noch vorhandene Elemente, die solche , wie z. B. `font` , zulassen , sind "völlig veraltet und dürfen nicht von Autoren verwendet werden" [1] .

HTML wird manchmal als Programmiersprache bezeichnet, hat aber keine Logik, also eine

Auszeichnungssprache . HTML-Tags sorgen für semantische Bedeutung und maschinelle Lesbarkeit des Inhalts der Seite.

Ein Element besteht normalerweise aus einem öffnenden Tag (`<element_name>`), einem schließenden Tag (`</element_name>`), der den von spitzen Klammern umgebenen Namen des Elements enthält und den Inhalt dazwischen: `<element_name>...content...</element_name>`

Es gibt einige HTML-Elemente, die weder ein schließendes Tag noch Inhalte haben. Diese werden als leere Elemente bezeichnet . Zu den ungültigen Elementen gehören `<img>` , `<meta>` , `<link>` und `<input>` .

Elementnamen können als beschreibende Schlüsselwörter für den Inhalt betrachtet werden, wie z. B. `video` , `audio` , `table` und `footer` .

Eine HTML-Seite kann aus potenziell Hunderten von Elementen bestehen, die dann von einem Webbrowser gelesen, interpretiert und in vom Menschen lesbaren oder hörbaren Inhalt auf dem Bildschirm dargestellt werden.

Für dieses Dokument ist es wichtig, den Unterschied zwischen Elementen und Tags zu beachten:

Elemente: `video` , `audio` , `table` , `footer`

Tags: `<video>` , `<audio>` , `<table>` , `<footer>` , `</html>` , `</body>`

Elementeinblick

Lassen Sie uns einen Tag aufschlüsseln ...

Das `<p>` -Tag repräsentiert einen gemeinsamen Absatz.

Elemente haben im Allgemeinen ein öffnendes Tag und ein schließendes Tag. Das öffnende Tag enthält den Namen des Elements in spitzen Klammern (`<p>`). Das schließende Tag ist identisch mit dem öffnenden Tag, wobei zwischen der öffnenden Klammer und dem Namen des Elements (`</p>`) ein Schrägstrich (`/`) eingefügt wird.

Der Inhalt kann dann zwischen diesen beiden Tags liegen: `<p>This is a simple paragraph.</p>` .

Eine einfache Seite erstellen

Das folgende HTML-Beispiel erstellt eine einfache Webseite "Hello World" .

HTML-Dateien können mit einem beliebigen Texteditor erstellt werden. Die Dateien müssen mit

der Erweiterung `.html` oder `.htm` [2] gespeichert werden, um als HTML-Dateien erkannt zu werden.

Einmal erstellt, kann diese Datei in einem beliebigen Webbrowser geöffnet werden.

```
<!DOCTYPE html>
<html lang="en">

    <head>
        <meta charset="UTF-8">
        <title>Hello!</title>
    </head>

    <body>
        <h1>Hello World!</h1>
        <p>This is a simple paragraph.</p>
    </body>

</html>
```

Einfacher Seitenumbruch

Dies sind die im Beispiel verwendeten Tags:

Etikett	Bedeutung
`<!DOCTYPE>`	Definiert die im Dokument verwendete HTML-Version. In diesem Fall ist es HTML5. Weitere Informationen finden Sie im Thema doctypes .
`<html>`	Öffnet die Seite. Nach dem schließenden Tag (`</html>`) sollte kein Markup erscheinen. Das `lang` Attribut deklariert die Hauptsprache der Seite mit den ISO-Sprachcodes (`en` für Englisch). Weitere Informationen finden Sie im Thema Inhaltssprache .
`<head>`	Öffnet das Kopfteil, die nicht im Hauptfenster des Browsers erscheinen , aber in erster Linie enthält Informationen *über* das HTML - Dokument, *Metadaten* genannt. Es kann auch Importe aus externen Stylesheets und Skripten enthalten. Das schließende Tag ist `</head>` .
`<meta>`	Liefert dem Browser einige Metadaten zum Dokument. Das `charset` deklariert die Zeichenkodierung . Moderne HTML-Dokumente sollten immer UTF-8 verwenden , obwohl dies nicht erforderlich ist. In HTML erfordert das `<meta>` -Tag kein schließendes Tag. Weitere Informationen finden Sie im Meta-Thema .

Etikett	Bedeutung
`<title>`	Der Titel der Seite. Text, der zwischen diesem Anfangs- und dem schließenden Tag (`</title>`) geschrieben wird, wird auf der Registerkarte der Seite oder in der Titelleiste des Browsers angezeigt.
`<body>`	Öffnet den Teil des Dokuments, der den Benutzern angezeigt wird, dh alle sichtbaren oder hörbaren Inhalte einer Seite. Nach dem schließenden Tag sollte kein Inhalt hinzugefügt werden `</body>` .
`<h1>`	Eine Überschrift der Ebene 1 für die Seite. Weitere Informationen finden Sie in den Überschriften .
`<p>`	Stellt einen gemeinsamen Textabschnitt dar.

---

1. ↑ HTML5, 11.2 Nichtkonforme Funktionen
2. ↑ `.htm` wird vom `.htm` Dateierweiterungslimit für DOS mit drei Zeichen geerbt.

Erste Schritte mit HTML online lesen: https://riptutorial.com/de/html/topic/217/erste-schritte-mit-html

Kapitel 2: Absätze

Einführung

Absätze sind das grundlegendste HTML-Element. In diesem Thema wird die Verwendung des Absatzelements in HTML erläutert und veranschaulicht.

Parameter

Säule	Säule
`<p>`	Definiert einen Absatz
` `	Fügt einen einzelnen Zeilenumbruch ein
`<pre>`	Definiert vorformatierten Text

Examples

HTML-Absätze

Das HTML-Element `<p>` definiert einen **Absatz** :

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Anzeige-

Sie können nicht sicher sein, wie HTML angezeigt wird.

Große oder kleine Bildschirme und skalierte Fenster führen zu unterschiedlichen Ergebnissen.

Mit HTML können Sie die Ausgabe nicht ändern, indem Sie Ihrem HTML-Code zusätzliche Leerzeichen oder zusätzliche Zeilen hinzufügen.

Der Browser entfernt zusätzliche Leerzeichen und zusätzliche Zeilen, wenn die Seite angezeigt wird:

```
    <p>This is          another     paragraph, extra spaces   will be   removed  by
browsers</p>
```

Absätze online lesen: https://riptutorial.com/de/html/topic/7997/absatze

Kapitel 3: Anker und Hyperlinks

Einführung

Ankertags werden im Allgemeinen zum Verknüpfen separater Webseiten verwendet. Sie können jedoch auch zum Verknüpfen zwischen verschiedenen Stellen in einem einzelnen Dokument verwendet werden, häufig im Inhaltsverzeichnis oder sogar zum Starten externer Anwendungen. In diesem Thema wird die Implementierung und Anwendung von HTML-Ankertags in verschiedenen Rollen erläutert.

Syntax

- `<a href="URL or anchor">Link Text</a>`

Parameter

Parameter	Einzelheiten
`href`	Gibt die Zieladresse an. Dies kann eine absolute oder relative URL oder der `name` eines Ankers sein. Eine absolute URL ist die vollständige URL einer Website wie http://example.com/ . Eine relative URL verweist auf ein anderes Verzeichnis und / oder Dokument in derselben Website, z. B. `/about-us/` verweist auf das Verzeichnis "about-us" im Stammverzeichnis (`/`). Wenn Sie auf ein anderes Verzeichnis zeigen, ohne das Dokument explizit anzugeben, geben Webserver normalerweise das Dokument "index.html" in diesem Verzeichnis zurück.
`hreflang`	Gibt die Sprache der Ressource an, die mit dem Attribut `href` verknüpft ist (die bei diesem Attribut vorhanden sein muss). Verwenden Sie Sprachwerte aus BCP 47 für HTML5 und RFC 1766 für HTML 4.
`rel`	Gibt die Beziehung zwischen dem aktuellen Dokument und dem verknüpften Dokument an. Für HTML5 müssen die Werte in der Spezifikation definiert oder im Microformats-Wiki registriert sein .
`target`	Gibt an, wo der Link geöffnet werden soll, z. B. in einem neuen Tab oder Fenster. Mögliche Werte sind `_blank` , `_self` , `_parent` , `_top` und `framename` (veraltet). Das Erzwingen eines solchen Verhaltens wird nicht empfohlen, da dadurch die Kontrolle des Benutzers über eine Website verletzt wird.
`title`	Gibt zusätzliche Informationen zu einer Verknüpfung an. Die Informationen werden meistens als QuickInfo-Text angezeigt, wenn der Cursor über den Link bewegt wird. Dieses Attribut ist nicht auf Links beschränkt, sondern kann für fast alle HTML-Tags verwendet werden.

Parameter	Einzelheiten
`download`	Gibt an, dass das Ziel heruntergeladen wird, wenn ein Benutzer auf den Hyperlink klickt. Der Wert des Attributs ist der Name der heruntergeladenen Datei. Es gibt keine Einschränkungen für die zulässigen Werte, und der Browser erkennt automatisch die richtige Dateierweiterung und fügt sie der Datei hinzu (.img, .pdf usw.). Wenn der Wert nicht angegeben wird, wird der ursprüngliche Dateiname verwendet.

Examples

Link zu einer anderen Site

Dies ist die grundlegende Verwendung des Elements `<a>` (**ein** nchor-Element) :

```
<a href="http://example.com/">Link to example.com</a>
```

Es erstellt einen Hyperlink zur URL `http://example.com/` wie durch das Attribut `href` (Hypertextreferenz) angegeben, mit dem Ankertext "Link to example.com". Es würde ungefähr so aussehen:

Link zu example.com

Um anzugeben, dass dieser Link zu einer externen Website führt, können Sie den Typ des `external` Links verwenden:

```
<a href="http://example.com/" rel="external">example site</a>
```

Sie können eine Verbindung zu einer Site herstellen, die ein anderes Protokoll als HTTP verwendet. Um beispielsweise eine Verbindung zu einer FTP-Site herzustellen, können Sie Folgendes tun:

```
<a href="ftp://example.com/">This could be a link to a FTP site</a>
```

In diesem Fall besteht der Unterschied darin, dass dieses Ankertag fordert, dass der Browser des Benutzers eine Verbindung zu `example.com` über das File Transfer Protocol (FTP) anstelle des Hypertext Transfer Protocol (HTTP) herstellt.

Dies könnte ein Link zu einer FTP-Site sein

Link in neuem Tab / Fenster öffnen

```
<a href="example.com" target="_blank">Text Here</a>
```

Das `target` gibt an, wo die Verknüpfung geöffnet werden soll. Durch die Einstellung auf `_blank` Sie

den Browser an, ihn in einer neuen Registerkarte oder einem neuen Fenster zu öffnen (je nach Präferenz des Benutzers).

> **SICHERHEIT VULNERABILITY WARNUNG!**
>
> Wenn Sie `target="_blank"` erhält die öffnende Site über JavaScript teilweise Zugriff auf das `window.opener` Objekt. `window.opener` Seite kann dann auf die `window.opener.location` *Ihrer* Seite zugreifen und diese ändern und möglicherweise Benutzer auf Malware- oder Phishing-Sites umleiten.
>
> Wenn Sie dies für Seiten verwenden, die Sie nicht steuern, fügen `rel="noopener"` Ihrem Link `rel="noopener"` , um zu verhindern, dass das `window.opener` Objekt mit der Anforderung gesendet wird.
>
> Derzeit unterstützt Firefox `noopener` , daher müssen Sie `rel="noopener noreferrer"` um einen maximalen Effekt zu erzielen.

Link zu einem Anker

Anker können verwendet werden, um zu bestimmten Tags auf einer HTML-Seite zu springen. Das `<a>` -Tag kann auf jedes Element zeigen, das ein `id` Attribut hat. Weitere Informationen zu IDs finden Sie in der Dokumentation zu Klassen und IDs . Anker werden meistens verwendet, um zu einem Unterabschnitt einer Seite zu springen, und werden in Verbindung mit Header-Tags verwendet.

Angenommen, Sie haben eine Seite (`page1.html`) zu vielen Themen erstellt:

```
<h2>First topic</h2>
<p>Content about the first topic</p>
<h2>Second topic</h2>
<p>Content about the second topic</p>
```

Wenn Sie mehrere Abschnitte haben, möchten Sie möglicherweise oben auf der Seite ein Inhaltsverzeichnis mit Quicklinks (oder Lesezeichen) zu bestimmten Abschnitten erstellen.

Wenn Sie Ihren Themen ein `id` Attribut zugewiesen haben, können Sie sie mit diesen verknüpfen

```
<h2 id="Topic1">First topic</h2>
<p>Content about the first topic</p>
<h2 id="Topic2">Second topic</h2>
<p>Content about the second topic</p>
```

Jetzt können Sie den Anker in Ihrem Inhaltsverzeichnis verwenden:

```
<h1>Table of Contents</h1>
    <a href='#Topic1'>Click to jump to the First Topic</a>
    <a href='#Topic2'>Click to jump to the Second Topic</a>
```

Diese Anker sind auch mit der Webseite verbunden, auf der sie sich befinden (`page1.html`). Sie können also von einer Seite zur anderen über die Site verweisen, indem Sie auf den Namen der

Seite *und des* Ankers verweisen.

```
 Remember, you can always <a href="page1.html#Topic1">look back in the First Topic</a> for supporting information.
```

Link, auf dem JavaScript ausgeführt wird

Verwenden Sie einfach das Protokoll `javascript:` :, um den Text als JavaScript auszuführen, anstatt ihn als normalen Link zu öffnen:

```
<a href="javascript:myFunction();">Run Code</a>
```

Sie können dasselbe auch mit dem Attribut `onclick` :

```
<a href="#" onclick="myFunction(); return false;">Run Code</a>
```

Die `return false;` ist erforderlich, um zu verhindern, dass Ihre Seite nach oben scrollen kann, wenn auf den Link zu `#` geklickt wird. Stellen Sie sicher, dass Sie den gesamten Code einschließen, den Sie zuvor ausführen möchten, da durch die Rückgabe die Ausführung des weiteren Codes beendet wird.

Bemerkenswert ist auch, dass Sie ein Ausrufezeichen einfügen können `!` nach dem Hashtag, um zu verhindern, dass die Seite nach oben scrollen kann. Dies funktioniert, weil bei einem ungültigen Slug der Link *nirgendwo* auf der Seite verschoben werden *kann* , da er das Element, auf das er verweist, nicht finden konnte (ein Element mit `id="!"`). Sie können auch nur einen ungültigen Slug (wie `#scrollsNowhere`) verwenden, um denselben Effekt zu erzielen. In diesem Fall `return false;` ist nicht nötig:

```
<a href="#!" onclick="myFunction();">Run Code</a>
```

> **Sollten Sie etwas davon verwenden?**
>
> Die Antwort lautet fast sicher ***nein*** . Das Ausführen von JavaScript inline mit diesem Element ist eine ziemlich schlechte Praxis. Verwenden Sie reine JavaScript-Lösungen, die das Element auf der Seite suchen und stattdessen eine Funktion daran binden. Auf eine Veranstaltung hören
>
> Überlegen Sie auch, ob es sich bei diesem Element wirklich um einen *Button* anstatt um einen *Link handelt* . Wenn ja, sollten Sie `<button>` .

Link zu einer Seite auf derselben Site

Sie können einen relativen Pfad verwenden , um auf Seiten derselben Website zu verlinken.

```
<a href="/example">Text Here</a>
```

Das obige Beispiel würde in die Datei geht `example` im Stammverzeichnis (`/`) des Servers.

Wenn sich dieser Link auf http://example.com befindet , können die folgenden beiden Links den Benutzer an denselben Speicherort bringen

```
<a href="/page">Text Here</a>
<a href="http://example.com/page">Text Here</a>
```

Beide oben gehen würden , um die `page` Datei im Stammverzeichnis von `example.com` .

Link, auf dem der E-Mail-Client ausgeführt wird

Grundlegende Verwendung

Wenn der Wert des `href` Attributs mit `mailto:` beginnt `mailto:` Es wird versucht, einen E-Mail-Client beim Klicken zu öffnen:

```
<a href="mailto:example@example.com">Send email</a>
```

Dadurch wird die E-Mail-Adresse `example@example.com` als Empfänger für die neu erstellte E-Mail angegeben.

Cc und Bcc

Sie können auch Adressen für cc- oder bcc-Empfänger mit der folgenden Syntax hinzufügen:

```
<a href="mailto:example@example.com?cc=john@example.com&bcc=jane@example.com">Send email</a>
```

Betreff und Haupttext

Sie können auch den Betreff und den Nachrichtentext für die neue E-Mail eingeben:

```
<a href="mailto:example@example.com?subject=Example+subject&body=Message+text">Send email</a>
```

Diese Werte müssen URL-kodiert sein .

Durch Klicken auf einen Link mit `mailto:` wird versucht, den von Ihrem Betriebssystem angegebenen Standard-E-Mail-Client zu öffnen, oder Sie werden gefragt, welchen Client Sie verwenden möchten. Nicht alle Optionen, die nach der Empfängeradresse angegeben sind, werden in allen E-Mail-Clients unterstützt.

Link, der eine Nummer wählt

Wenn der Wert des `href` Attributs mit `tel:` beginnt, wählt das Gerät die Nummer, wenn Sie darauf klicken. Dies funktioniert auf mobilen Geräten oder auf Computern / Tablets, auf denen Software - wie Skype oder FaceTime - ausgeführt wird, die telefonieren kann.

```
<a href="tel:11234567890">Call us</a>
```

Die meisten Geräte und Programme fordern den Benutzer auf, die zu wählende Nummer zu bestätigen.

Anker und Hyperlinks online lesen: https://riptutorial.com/de/html/topic/254/anker-und-hyperlinks

Kapitel 4: ARIE

Syntax

- Arie-Live
- arienrelevant
- Arie-Autovervollständigung
- aria-geprüft
- Aria-Behinderte
- Arie erweitert
- aria-haspopup
- Arie versteckt
- aria-ungültig
- Aria-Label
- Aria-Ebene
- Arie-Multiline
- Arie-multiselektierbar
- Arienorientierung
- ariengepresst
- aria-readonly
- Arie erforderlich
- Arie ausgewählt
- aria-sort
- Aria-ValueMax
- aria-valuemin
- Aria-Valuenow
- Aria-Wertetext
- Aria-Atomic
- Arie beschäftigt
- aria-dropeffekt
- Arie gezerrt
- Aria-aktivierterAnhänger
- Aria-Kontrollen
- Arie-beschriebenvon
- aria-flowto
- Arie beschriftetvon
- Aria besitzt
- Aria-Posinset
- aria-setsize

Bemerkungen

ARIA ist eine Spezifikation für die semantische Beschreibung von Rich-Web-Anwendungen. Wenn Sie sich an ARIA-Standards halten, können Benutzer, die assistive Technologien nutzen (z. B.

Bildschirmleser), den Zugriff auf Ihre Inhalte verbessern.

Examples

role = "alert"

Eine Nachricht mit wichtigen und normalerweise zeitkritischen Informationen.

```
<div role="alert" aria-live="assertive">Your session will expire in 60 seconds.</div>
```

> Beachten Sie, dass ich gleichzeitig sowohl `role="alert"` als auch `aria-live="assertive"` habe. Dies sind auch Attribute, aber einige Screenreader unterstützen nur das eine oder das andere. Durch die gleichzeitige Verwendung beider maximieren wir daher die Chancen, dass die Live-Region wie erwartet funktioniert.
>
> Quelle - Heydon Pickering 'Einige praktische ARIA-Beispiele'

role = "alertdialog"

Ein Typ eines Dialogfelds, das eine Warnmeldung enthält, wobei der anfängliche Fokus auf ein Element innerhalb des Dialogfelds gerichtet ist.

```
<div role="alertdialog">
  <h1>Warning</h1>
  <div role="alert">Your session will expire in 60 seconds.</div>
</div>
```

role = "Anwendung"

Eine als Webanwendung deklarierte Region im Gegensatz zu einem Webdokument. In diesem Beispiel handelt es sich bei der Anwendung um einen einfachen Rechner, der möglicherweise zwei Zahlen addiert.

```
<div role="application">
  <h1>Calculator</h1>
  <input id="num1" type="text"> + <input id="num2" type="text"> =
  <span id="result"></span>
</div>
```

role = "Artikel"

Ein Abschnitt einer Seite, der aus einer Komposition besteht, die einen unabhängigen Teil eines Dokuments, einer Seite oder einer Site bildet.

> Das Festlegen einer ARIA-Rolle und / oder eines aria- * -Attributs, die der standardmäßigen impliziten ARIA-Semantik entsprechen, ist nicht erforderlich und wird nicht empfohlen, da diese Eigenschaften bereits vom Browser festgelegt werden.

```
<article>
  <h1>My first article</h1>
  <p>Lorem ipsum...</p>
</article>
```

Sie würden `role=article` für nicht semantische Elemente verwenden (nicht empfohlen, ungültig).

```
<div role="article">
  <h1>My first article</h1>
  <p>Lorem ipsum...</p>
</div>
```

W3C Eintrag für `role=article`

role = "banner"

Eine Region, die eher seitenorientierten Inhalt als seitenabhängigen Inhalt enthält.

```
<div role="banner">
  <h1>My Site</h1>

  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="/about">About</a></li>
    <li><a href="/contact">Contact</a></li>
  </ul>
</div>
```

role = "button"

Eine Eingabe, die vom Benutzer ausgelöste Aktionen beim Klicken oder Drücken ermöglicht

```
<button role="button">Add</button>
```

Rolle = "Zelle"

Eine Zelle in einem Tabellencontainer.

```
<table>
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <td role="cell">95</td>
    <td role="cell">14</td>
    <td role="cell">25</td>
  </tbody>
</table>
```

role = "Ankreuzfeld"

Eine überprüfbare Eingabe mit drei möglichen Werten: true, false oder mixed.

```
<p>
  <input type="checkbox" role="checkbox" aria-checked="false">
  I agree to the terms
</p>
```

role = "columnheader"

Eine Zelle, die Header-Informationen für eine Spalte enthält.

```
<table role="grid">
  <thead>
    <tr>
      <th role="columnheader">Day 1</th>
      <th role="columnheader">Day 2</th>
      <th role="columnheader">Day 3</th>
    </tr>
  </thead>
  <tbody>
    <!-- etc -->
  </tbody>
<table>
```

role = "Kombinationsfeld"

Eine Präsentation eines select; In der Regel ähnlich wie bei einem Textfeld, in das Benutzer zuvor eine Option auswählen können, oder einen beliebigen Text als neuen Eintrag in die Liste.

```
<input type="text" role="combobox" aria-expanded="false">
```

Normalerweise verwenden Sie JavaScript, um den Rest der Typhead- oder Listenauswahlfunktionalität zu erstellen.

Rolle = "komplementär"

Ein unterstützender Abschnitt des Dokuments, der auf einer ähnlichen Ebene in der DOM-Hierarchie den Hauptinhalt ergänzen soll, bleibt jedoch bedeutsam, wenn er vom Hauptinhalt getrennt wird.

```
<div role="complementary">
  <h2>More Articles</h2>

  <ul>
    <!-- etc -->
  </ul>
</div>
```

role = "contentinfo"

Ein großer wahrnehmbarer Bereich, der Informationen zum übergeordneten Dokument enthält.

```
<p role="contentinfo">
  Author: Albert Einstein<br>
  Published: August 15, 1940
</p>
```

Rolle = "Definition"

Eine Definition eines Begriffs oder Begriffs.

```
<span role="term" aria-labelledby="def1">Love</span>
<span id="def1" role="definition">an intense feeling of deep affection.</span>
```

role = "dialog"

Ein Dialog ist ein Anwendungsfenster, das die aktuelle Verarbeitung einer Anwendung unterbricht, um den Benutzer zur Eingabe von Informationen oder zur Antwort aufzufordern.

```
<div role="dialog">
  <p>Are you sure?</p>
  <button role="button">Yes</button>
  <button role="button">No</button>
</div>
```

role = "verzeichnis"

Eine Liste von Verweisen auf Mitglieder einer Gruppe, z. B. ein statisches Inhaltsverzeichnis.

```
<ul role="directory">
  <li><a href="/chapter-1">Chapter 1</a></li>
  <li><a href="/chapter-2">Chapter 2</a></li>
  <li><a href="/chapter-3">Chapter 3</a></li>
</ul>
```

role = "Dokument"

Eine Region, die verwandte Informationen enthält, die im Gegensatz zu einer Webanwendung als Dokumentinhalt deklariert sind.

```
<div role="document">
  <h1>The Life of Albert Einstein</h1>
  <p>Lorem ipsum...</p>
</div>
```

role = "form"

Ein Markierungsbereich, der eine Sammlung von Elementen und Objekten enthält, die insgesamt zu einem Formular kombiniert werden.

Die Verwendung des semantisch korrekten HTML-Elements `<form>` impliziert die standardmäßige

ARIA-Semantik. Die Bedeutung von `role=form` ist nicht erforderlich, da Sie keine kontrastierende Rolle auf ein bereits semantisches Element anwenden sollten, da durch das Hinzufügen einer Rolle die native Semantik eines Elements überschrieben wird.

> Das Festlegen einer ARIA-Rolle und / oder eines aria- * -Attributs, die der standardmäßigen impliziten ARIA-Semantik entsprechen, ist nicht erforderlich und wird nicht empfohlen, da diese Eigenschaften bereits vom Browser festgelegt werden.

```
<form action="">
  <fieldset>
    <legend>Login form</legend>
    <div>
      <label for="username">Your username</label>
      <input type="text" id="username" aria-describedby="username-tip" required />
      <div role="tooltip" id="username-tip">Your username is your email address</div>
    </div>
    <div>
      <label for="password">Your password</label>
      <input type="text" id="password" aria-describedby="password-tip" required />
      <div role="tooltip" id="password-tip">Was emailed to you when you signed up</div>
    </div>
  </fieldset>
</form>
```

Sie würden `role=form` für nicht semantische Elemente verwenden (nicht empfohlen, ungültig).

```
<div role=form>
  <input type="email" placeholder="Your email address">
  <button>Sign up</button>
</div>
```

role = "grid"

Ein Raster ist ein interaktives Steuerelement, das Tabellendaten enthält, die in Zeilen und Spalten wie eine Tabelle angeordnet sind.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <!-- etc -->
  </tbody>
</table>
```

role = "gridcell"

Eine Zelle in einem Gitter oder Baumgrill.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
```

```
  <tbody>
    <tr>
      <td role="gridcell">17</td>
      <td role="gridcell">64</td>
      <td role="gridcell">18</td>
    </tr>
  </tbody>
<table>
```

Rolle = "Gruppe"

Eine Gruppe von Benutzeroberflächenobjekten, die von assistiven Technologien nicht in eine Seitenübersicht oder ein Inhaltsverzeichnis einbezogen werden sollen.

```
<div role="group">
  <button role"button">Previous</button>
  <button role"button">Next</button>
</div>
```

role = "Überschrift"

Eine Überschrift für einen Abschnitt der Seite.

```
<h1 role="heading">Introduction</h1>
<p>Lorem ipsum...</p>
```

role = "img"

Ein Container für eine Sammlung von Elementen, die ein Bild bilden.

```
<figure role="img">
  <img alt="A cute cat." src="albert.jpg">
  <figcaption>This is my cat, Albert.</figcaption>
<figure>
```

role = "link"

Eine interaktive Referenz zu einer internen oder externen Ressource, die bei ihrer Aktivierung den Benutzeragenten dazu bringt, zu dieser Ressource zu navigieren.

> In den meisten Fällen ist das Festlegen einer ARIA-Rolle und / oder eines aria- * -Attributs, die der standardmäßigen impliziten ARIA-Semantik entsprechen, nicht erforderlich und wird nicht empfohlen, da diese Eigenschaften bereits vom Browser festgelegt werden.
>
> Quelle - https://www.w3.org/TR/html5/dom.html#aria-usage-note

role = "list"

Eine Gruppe nicht interaktiver Listenelemente.

```
<ul role="list">
  <li role="listitem">One</li>
  <li role="listitem">Two</li>
  <li role="listitem">Three</li>
</ul>
```

role = "Listbox"

Ein Widget, mit dem der Benutzer ein oder mehrere Elemente aus einer Auswahlliste auswählen kann.

```
<ul role="listbox">
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
</ul>
```

Normalerweise würden Sie JavaScript verwenden, um die Funktion zur Mehrfachauswahl zu erstellen.

role = "listitem"

Ein einzelnes Element in einer Liste oder einem Verzeichnis.

```
<ul role="list">
  <li role="listitem">One</li>
  <li role="listitem">Two</li>
  <li role="listitem">Three</li>
</ul>
```

role = "log"

Eine Art von Live-Region, in der neue Informationen in sinnvoller Reihenfolge hinzugefügt werden und alte Informationen möglicherweise verschwinden.

```
<ul role="log">
  <li>User 1 logged in.</li>
  <li>User 2 logged in.</li>
  <li>User 1 logged out.</li>
</ul>
```

role = "main"

Der Hauptinhalt eines Dokuments.

```
<!-- header & nav here -->
<div role="main">
  <p>Lorem ipsum...</p>
</div>
<!-- footer here -->
```

Rolle = "Laufschrift"

Eine Art von Live-Region, in der sich nicht wesentliche Informationen häufig ändern.

```
<ul role="marquee">
  <li>Dow +0.26%</li>
  <li>Nasdaq +0.54%</li>
  <li>S&amp;amp;P +0.44%</li>
</ul>
```

role = "math"

Inhalt, der einen mathematischen Ausdruck darstellt.

```
<img role="math" alt="y=mx+b" src="slope.png">
```

role = "Menü"

Ein Widget-Typ, der dem Benutzer eine Auswahlliste bietet.

```
<ul role="menu">
  <li role="menuitem">New</li>
  <li role="menuitem">Open</li>
  <li role="menuitem">Save</li>
  <li role="menuitem">Close</li>
</ul>
```

Rolle = "Menüleiste"

Eine Präsentation des Menüs, die normalerweise sichtbar bleibt und normalerweise horizontal angezeigt wird.

```
<ul role="menubar">
  <li role="menuitem">File</li>
  <li role="menuitem">Edit</li>
  <li role="menuitem">View</li>
  <li role="menuitem">Help</li>
</ul>
```

role = "menuitem"

Eine Option in einer Gruppe von Optionen, die in einem Menü oder einer Menüleiste enthalten sind.

```
<ul role="menubar">
  <li role="menuitem">File</li>
  <li role="menuitem">Edit</li>
  <li role="menuitem">View</li>
  <li role="menuitem">Help</li>
</ul>
```

role = "menuitemcheckbox"

Ein überprüfbares Menü, das drei mögliche Werte hat: wahr, falsch oder gemischt.

```
<ul role="menu">
  <li role="menuitem">Console</li>
  <li role="menuitem">Layout</li>
  <li role="menuitemcheckbox" aria-checked="true">Word wrap</li>
</ul>
```

Rolle = "Menuitemradio"

Ein überprüfbares Menuitem in einer Gruppe von Menuitemradio-Rollen, von denen nur eine einzeln geprüft werden kann.

```
<ul role="menu">
  <li role="menuitemradio" aria-checked="true">Left</li>
  <li role="menuitemradio" aria-checked="false">Center</li>
  <li role="menuitemradio" aria-checked="false">Right</li>
</ul>
```

role = "navigation"

Eine Sammlung von Navigationselementen (normalerweise Links) zum Navigieren im Dokument oder verwandten Dokumenten.

```
<ul role="navigation">
  <li><a href="/">Home</a></li>
  <li><a href="/about">About</a></li>
  <li><a href="/contact">Contact</a></li>
</ul>
```

role = "note"

Ein Abschnitt, dessen Inhalt dem Hauptinhalt der Ressource parenthetisch oder ergänzend ist.

```
<p>Lorem ipsum...</p>
<p>Lorem ipsum...</p>
<p role="note">Lorem ipsum...</p>
```

role = "Option"

Ein auswählbares Element in einer Auswahlliste.

```
<ul role="listbox">
  <li role="option">Option 1</li>
  <li role="option">Option 2</li>
  <li role="option">Option 3</li>
</ul>
```

Rolle = "Präsentation"

Ein Element, dessen implizite systemeigene Rollensemantik der Zugriffs-API nicht zugeordnet wird

```
<div style="float:left;">Some content on the left.</div>
<div style="float:right;">Some content on the right</div>
<div role="presentation" style="clear:both;"></div> <!-- Only used to clear floats -->
```

role = "progressbar"

Ein Element, das den Fortschrittsstatus für Aufgaben anzeigt, die lange dauern.

```
<progress role="progressbar" value="25" max="100">25%</progress>
```

Rolle = "Radio"

Eine überprüfbare Eingabe in einer Gruppe von Funkrollen, von denen jeweils nur eine überprüft werden kann.

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> One<br>
  <input role="radio" type="radio" aria-checked="false"> Two<br>
  <input role="radio" type="radio" aria-checked="false"> Three
</div>
```

Rolle = "Region"

Ein großer wahrnehmbarer Abschnitt einer Webseite oder eines Dokuments, von dem der Autor hält, dass er wichtig genug ist, um in eine Seitenzusammenfassung oder ein Inhaltsverzeichnis aufgenommen zu werden, beispielsweise einen Bereich der Seite mit Live-Sportereignisstatistiken.

```
<div role="region">
  Home team: 4<br>
  Away team: 2
</div>
```

Rolle = "Radiogruppe"

Eine Gruppe von Optionsfeldern.

```
<div role="radiogroup">
  <input role="radio" type="radio" aria-checked="true"> One<br>
  <input role="radio" type="radio" aria-checked="false"> Two<br>
  <input role="radio" type="radio" aria-checked="false"> Three
</div>
```

role = "Reihe"

Eine Reihe von Zellen in einem Tabellencontainer.

```
<table>
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <tr role="row">
      <!-- etc -->
    </tr>
  </tbody>
</table>
```

role = "Zeilengruppe"

Eine Gruppe mit einem oder mehreren Zeilenelementen in einem Raster.

```
<table>
  <thead role="rowgroup">
    <!-- etc -->
  </thead>
  <tbody role="rowgroup">
    <!-- etc -->
  </tbody>
</table>
```

role = "Rowheader"

Eine Zelle, die Kopfzeileninformationen für eine Zeile in einem Raster enthält.

```
<table role="grid">
  <thead>
    <!-- etc -->
  </thead>
  <tbody>
    <tr>
      <th role="rowheader">Day 1</th>
      <td>65</td>
    </tr>
    <tr>
      <th role="rowheader">Day 2</th>
      <td>74</td>
    </tr>
  </tbody>
</table>
```

role = "scrollbar"

Ein grafisches Objekt, das den Bildlauf des Inhalts innerhalb eines Anzeigebereichs steuert, unabhängig davon, ob der Inhalt vollständig im Anzeigebereich angezeigt wird.

```
<div id="content1">Lorem ipsum...</div>
<div
  role="scrollbar"
  aria-controls="content1"
  aria-orientation="vertical"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
    <div class="scrollhandle"></div>
</div>
```

role = "Suche"

Ein Markierungsbereich, der eine Sammlung von Elementen und Objekten enthält, die zusammen eine Suchfunktion erstellen.

```
<div role="search">
  <input role="searchbox" type="text">
  <button role="button">Search</button>
</div>
```

role = "Suchfeld"

Ein Textfeldtyp, der zum Angeben von Suchkriterien vorgesehen ist.

```
<div role="search">
  <input role="searchbox" type="text">
  <button role="button">Search</button>
</div>
```

role = "Trennzeichen"

Ein Teiler, der Inhaltsbereiche oder Gruppen von Menüelementen voneinander trennt und unterscheidet.

```
<p>Lorem ipsum...</p>
<hr role="separator">
<p>Lorem ipsum...</p>
```

Rolle = "Schieberegler"

Eine Benutzereingabe, bei der der Benutzer einen Wert aus einem bestimmten Bereich auswählt.

```
<div
  role="slider"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25">
    <div class="sliderhandle"></div>
</div>
```

role = "Drehfeld"

Eine Form des Bereichs, von der der Benutzer unter diskreten Auswahlmöglichkeiten auswählen muss.

```
<input
  role="spinbutton"
  aria-valuemax="100"
  aria-valuemin="0"
  aria-valuenow="25"
  type="number"
  value="25">
```

role = "status"

Ein Container, dessen Inhalt Empfehlungen für den Benutzer ist, aber nicht wichtig genug ist, um eine Warnung zu rechtfertigen, die häufig, aber nicht notwendigerweise als Statusleiste angezeigt wird.

```
<div role="status">Online</div>
```

role = "wechseln"

Eine Art Kontrollkästchen, das Ein / Aus-Werte im Gegensatz zu aktivierten / nicht geprüften Werten darstellt

```
<select role="switch" aria-checked="false">
  <option>On</option>
  <option selected>Off</option>
</select>
```

role = "tab"

Ein Gruppierungsetikett, das einen Mechanismus zum Auswählen des Registerkarteninhalts bereitstellt, der dem Benutzer angezeigt werden soll

```
<ul role="tablist">
  <li role="tab">Introduction</li>
  <li role="tab">Chapter 1</li>
  <li role="tab">Chapter 2</li>
</ul>
```

role = "table"

Ein Abschnitt, der Daten enthält, die in Zeilen und Spalten angeordnet sind. Die Tabellenrolle ist für tabellarische Container gedacht, die nicht interaktiv sind.

```
<table role="table">
  <thead>
```

```
    <!-- etc -->
  </thead>
  <tbody>
    <!-- etc -->
  </tbody>
</table>
```

role = "tablist"

Eine Liste von Tabulatorelementen, die Verweise auf Tabulatorelemente sind.

```
<ul role="tablist">
   <li role="tab">Introduction</li>
   <li role="tab">Chapter 1</li>
   <li role="tab">Chapter 2</li>
</ul>
```

role = "tabpanel"

Ein Container für die Ressourcen, die einer Registerkarte zugeordnet sind, wobei jede Registerkarte in einer Tablist enthalten ist.

```
<ul role="tablist">
  <li role="tab">Introduction</li>
  <li role="tab">Chapter 1</li>
  <li role="tab">Chapter 2</li>
</ul>
<div role="tabpanel">
  <!-- etc -->
</div>
```

role = "Textfeld"

Eingabe, die Freiformtext als Wert zulässt.

```
<textarea role="textbox"></textarea>
```

role = "Timer"

Eine Art von Live-Region, die einen numerischen Zähler enthält, der die abgelaufene Zeit ab einem Startpunkt oder die bis zu einem Endpunkt verbleibende Zeit angibt.

```
<p>
  <span role="timer">60</span> seconds remaining.
</p>
```

role = "Werkzeugleiste"

Eine Sammlung häufig verwendeter Funktionstasten, die in kompakter visueller Form dargestellt werden.

```
<ul role="toolbar">
  <li><img alt="New" src="new.png"></li>
  <li><img alt="Open" src="open.png"></li>
  <li><img alt="Save" src="save.png"></li>
  <li><img alt="Close" src="close.png"></li>
</ul>
```

role = "Tooltip"

Ein Kontext-Popup, das eine Beschreibung für ein Element anzeigt.

```
<span aria-describedby="slopedesc">Slope</span>
<div role="tooltip" id="slopedesc">y=mx+b</div>
```

Normalerweise wird der Tooltip ausgeblendet. Bei Verwendung von JavaScript wird der Tooltip nach einer Verzögerung angezeigt, wenn der Benutzer über das von ihm beschriebene Element fährt.

role = "Baum"

Ein Listentyp, der untergeordnete verschachtelte Gruppen enthalten kann, die reduziert und erweitert werden können.

```
<ul role="tree">
  <li role="treeitem">
    Part 1
    <ul>
      <li role="treeitem">Chapter 1</li>
      <li role="treeitem">Chapter 2</li>
      <li role="treeitem">Chapter 3</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 2
    <ul>
      <li role="treeitem">Chapter 4</li>
      <li role="treeitem">Chapter 5</li>
      <li role="treeitem">Chapter 6</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 3
    <ul>
      <li role="treeitem">Chapter 7</li>
      <li role="treeitem">Chapter 8</li>
      <li role="treeitem">Chapter 9</li>
    </ul>
  </li>
</ul>
```

role = "treegrid"

Ein Raster, dessen Zeilen auf dieselbe Weise wie für einen Baum erweitert und reduziert werden können

role = "treeitem"

Ein Optionselement eines Baums. Dies ist ein Element innerhalb einer Baumstruktur, das erweitert oder reduziert werden kann, wenn es eine Unterebenengruppe von Baumstrukturen enthält.

```
<ul role="tree">
  <li role="treeitem">
    Part 1
    <ul>
      <li role="treeitem">Chapter 1</li>
      <li role="treeitem">Chapter 2</li>
      <li role="treeitem">Chapter 3</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 2
    <ul>
      <li role="treeitem">Chapter 4</li>
      <li role="treeitem">Chapter 5</li>
      <li role="treeitem">Chapter 6</li>
    </ul>
  </li>
  <li role="treeitem">
    Part 3
    <ul>
      <li role="treeitem">Chapter 7</li>
      <li role="treeitem">Chapter 8</li>
      <li role="treeitem">Chapter 9</li>
    </ul>
  </li>
</ul>
```

ARIE online lesen: https://riptutorial.com/de/html/topic/2734/arie

Kapitel 5: Ausgabeelement

Parameter

Attribut	Beschreibung
Global	Attribute, die für jedes HTML5-Element verfügbar sind. Umfassende Dokumentation dieser Attribute finden Sie unter MDN Global Attribute
Name	Eine Zeichenfolge, die den Namen einer Ausgabe darstellt. Als Formularelement kann die Ausgabe über die Eigenschaft `document.forms` anhand ihres Namens referenziert werden. Dieses Attribut wird auch zum Erfassen von Werten in einem Formularsenden verwendet.
zum	Eine durch Leerzeichen getrennte Liste von Formularelement-IDs (z. B. `<inputs id="inp1"> for value is "inp1"`), für die die Ausgabe Berechnungen anzeigen soll.
bilden	Eine Zeichenfolge, die das `<form>` , das der Ausgabe zugeordnet ist. Wenn die Ausgabe tatsächlich außerhalb dem ist `<form>` Dieses Attribut wird dafür sorgen , dass die Ausgabe gehört noch die `<form>` und vorbehaltlich den Sammlungen und legt die `<form>` .

Examples

Ausgabeelement mit for- und Formularattributen

In der folgenden Demo werden die Attribute `[for]` und `[form]` eines `<output>` -Elements verwendet. Beachten Sie, dass `<output>` **JavaScript benötigt** , um funktionieren zu können. Inline-JavaScript wird häufig in Formularen verwendet, wie dieses Beispiel zeigt. Obwohl die `<input>` -Elemente `type="number"` , sind ihre `value` keine Zahlen, sondern Text. Wenn Sie also die Berechnung des `value` s benötigen, müssen Sie jeden `value` mithilfe von Methoden wie `parseInt()` , `parseFloat()` , `Number()` usw. in eine Zahl konvertieren.

Live Demo

```
    <!--form1 will collect the values of in1 and in2 on 'input' event.-->
    <!--out1 value will be the sum of in1 and in2 values.-->

<form id="form1" name="form1" oninput="out1.value = parseInt(in1.value, 10) +
parseInt(in2.value, 10)">

  <fieldset>

    <legend>Output Example</legend>

      <input type="number" id="in1" name="in1" value="0">
    <br/>
```

```
    +
    <input type="number" id="in2" name="in2" value="0">

  </fieldset>

</form>

<!--[for] attribute enables out1 to display calculations for in1 and in2.-->
<!--[form] attribute designates form1 as the form owner of out1 even if it isn't a
descendant.-->

<output name="out1" for="in1 in2" form="form1">0</output>
```

Ausgabeelement mit Attributen

```
<output name="out1" form="form1" for="inp1 inp2"></output>
```

Ausgabeelement online lesen: https://riptutorial.com/de/html/topic/723/ausgabeelement

Kapitel 6: Auswahlmenü-Steuerelemente

Syntax

- `<select name=""></select>`
- `<datalist id=""></datalist>`
- `<optgroup label="Option Group"></optgroup>`
- `<option value="">Option</option>`

Examples

Wählen Sie Menü

Das Element `<select>` erzeugt ein Dropdown-Menü, aus dem der Benutzer eine Option auswählen kann.

```
<select name="">
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
  <option value="4">Four</option>
</select>
```

Größe ändern

Sie können die Größe des Auswahlmenüs mit dem `size` ändern. Bei einer Größe von 0 oder 1 wird das Standard-Dropdown-Menü angezeigt. Bei einer Größe von mehr als 1 wird das Dropdown-Menü in ein Feld umgewandelt, in dem so viele Zeilen angezeigt werden, wobei pro Zeile eine Option und eine Bildlaufleiste zum Blättern durch die verfügbaren Optionen angezeigt werden.

```
<select name="" size="4"></select>
```

Multioptionsauswahlmenüs

Standardmäßig können Benutzer nur eine einzige Option auswählen. Durch das Hinzufügen des `multiple` Attributs können Benutzer mehrere Optionen gleichzeitig auswählen und alle ausgewählten Optionen mit dem Formular übermitteln. Durch die Verwendung des Attributs `multiple` wird das Dropdown-Menü automatisch in eine Box umgewandelt, als wäre eine Größe definiert. In diesem Fall wird die Standardgröße durch den von Ihnen verwendeten Browser festgelegt. Es ist nicht möglich, sie wieder in ein Dropdown-Menü zu ändern, während mehrere Optionen ausgewählt werden.

```
<select name="" multiple></select>
```

Bei Verwendung des `multiple` Attributs besteht ein Unterschied zwischen der Verwendung von 0

und 1 für die Größe, wohingegen kein Unterschied besteht, wenn das Attribut nicht verwendet wird. Die Verwendung von 0 bewirkt, dass sich der Browser in der von ihm programmierten Standardeinstellung verhält. Bei Verwendung von 1 wird die Größe des resultierenden Felds explizit auf nur eine Zeile hoch festgelegt.

Optionsgruppen

Sie können Ihre Optionen innerhalb eines Auswahlmenüs ordentlich gruppieren, um ein strukturierteres Layout in einer langen Liste von Optionen bereitzustellen, indem Sie das Element `<optgroup>` verwenden.

Die Syntax ist sehr einfach, indem einfach das Element mit einem `label` wird, um den Titel für die Gruppe zu identifizieren und null oder mehr Optionen zu enthalten, die innerhalb dieser Gruppe stehen sollten.

```
<select name="">
  <option value="milk">Milk</option>
  <optgroup label="Fruits">
    <option value="banana">Bananas</option>
    <option value="strawberry">Strawberries</option>
  </optgroup>
  <optgroup label="Vegetables" disabled>
    <option value="carrot">Carrots</option>
    <option value="zucchini">Zucchini</option>
  </optgroup>
</select>
```

Bei der Verwendung von Optionsgruppen müssen nicht alle Optionen in einer Gruppe enthalten sein. Durch Deaktivieren einer Optionsgruppe werden auch alle Optionen in der Gruppe deaktiviert, und es ist nicht möglich, eine einzelne Option in einer deaktivierten Gruppe manuell erneut zu aktivieren.

Optionen

Die Optionen innerhalb eines Auswahlmenüs werden vom Benutzer ausgewählt. Die normale Syntax für eine Option lautet wie folgt:

```
<option>Some Option</option>
```

Beachten Sie jedoch, dass der Text im `<option>` -Element selbst nicht immer verwendet wird und im Wesentlichen der Standardwert für Attribute ist, die nicht angegeben sind.

Die Attribute, die das tatsächliche Aussehen und die Funktion der Option steuern, sind `value` und `label` . Die Beschriftung stellt den Text dar, der im Dropdown-Menü angezeigt wird (was Sie gerade betrachten und auf das Sie klicken werden, um es auszuwählen). Der Wert stellt den Text dar, der zusammen mit dem Formular gesendet wird. Wenn einer dieser Werte ausgelassen wird, wird stattdessen der Text innerhalb des Elements als Wert verwendet. So könnte das Beispiel, das wir oben gegeben haben, darauf "erweitert" werden:

```
<option label="Some Option" value="Some Option">
```

Beachten Sie die Auslassung des Innentextes und des End-Tags, die nicht unbedingt erforderlich sind, um eine Option innerhalb des Menüs zu erstellen. Wenn sie enthalten sind, wird der Innentext ignoriert, da beide Attribute bereits angegeben wurden und der Text nicht benötigt wird. Allerdings werden Sie wahrscheinlich nicht viele Leute sehen, die sie so schreiben. Die gebräuchlichste Art und Weise, in der geschrieben wird, ist ein Wert, der an den Server gesendet wird, zusammen mit dem Innentext, der schließlich zum Label-Attribut wird:

```
<option value="option1">Some Option</option>
```

Standardmäßig eine Option auswählen

Sie können auch eine bestimmte Option angeben, die standardmäßig im Menü `selected` werden soll, indem Sie das `selected` Attribut daran anhängen. Wenn im Menü keine Option als ausgewählt ausgewählt ist, wird beim Rendern standardmäßig die erste Option im Menü ausgewählt. Wenn mehr als eine Option mit dem `selected` Attribut verknüpft ist, ist die letzte im Menü mit dem Attribut vorhandene Option die standardmäßig ausgewählte.

```
<option value="option1" selected>Some option</option>
```

Wenn Sie das Attribut in einem Auswahlmenü mit mehreren Optionen verwenden, werden standardmäßig alle Optionen mit dem Attribut ausgewählt. Wenn keine Optionen über das Attribut verfügen, werden keine Optionen ausgewählt.

```
<select multiple>
  <option value="option1" selected>Some option</option>
  <option value="option2" selected>Some option</option>
</select>
```

Datenhändler

Das `<datalist>` -Tag gibt eine Liste vordefinierter Optionen für ein `<input>` -Element an. Es bietet eine "Autovervollständigung" für `<input>` -Elemente. Die Benutzer sehen beim Schreiben eine Dropdown-Liste mit Optionen.

```
<input list="Languages">

<datalist id="Languages">
  <option value="PHP">
  <option value="Perl">
  <option value="Python">
  <option value="Ruby">
  <option value="C+">
</datalist>
```

Browser-Unterstützung

Chrom	Kante	Mozilla	Safari	Oper
20,0	10,0	4,0	Nicht unterstützt	9,0

Auswahlmenü-Steuerelemente online lesen:
https://riptutorial.com/de/html/topic/722/auswahlmenu-steuerelemente

Kapitel 7: Bemerkungen

Einführung

Ähnlich wie andere Programmier-, Markup- und Markdown-Sprachen enthalten Kommentare in HTML andere Entwickler entwicklungsspezifische Informationen, ohne die Benutzeroberfläche zu beeinflussen. Im Gegensatz zu anderen Sprachen können HTML-Kommentare jedoch nur zum Angeben von HTML-Elementen für Internet Explorer verwendet werden. In diesem Thema wird erläutert, wie HTML-Kommentare und ihre funktionalen Anwendungen geschrieben werden.

Syntax

- `<!-- Comment text -->`

Bemerkungen

Alles, das mit `<!--` beginnt und mit `-->` endet, ist ein Kommentar. Kommentare dürfen nicht zwei benachbarte Striche (`--`) enthalten und müssen mit genau zwei Strichen enden (dh `--->` ist nicht korrekt).

Kommentare sind auf einer Webseite nicht sichtbar und können nicht mit CSS gestaltet werden. Sie können vom Entwickler der Seite verwendet werden, um Notizen im HTML-Code zu erstellen oder bestimmte Inhalte während der Entwicklung auszublenden.

Bei dynamischen oder interaktiven Seiten erfolgt das Ausblenden und Anzeigen von Inhalten mit JavaScript und CSS und nicht mit HTML-Kommentaren.

Mit JavaScript kann der Inhalt von HTML-Kommentarknoten abgerufen werden. Diese Knoten können dynamisch erstellt, hinzugefügt und aus dem Dokument entfernt werden. Dies hat jedoch keinen Einfluss auf die Darstellung der Seite.

Da HTML-Kommentare Teil des Quellcodes der Seite sind, werden sie zusammen mit dem Rest der Seite in den Browser heruntergeladen. Der Quellcode kann normalerweise mit der Menüoption des Webbrowsers "Quelle anzeigen" oder "Seitenquelle anzeigen" angezeigt werden.

Examples

Kommentare erstellen

HTML-Kommentare können verwendet werden, um sich oder anderen Entwicklern Notizen zu einem bestimmten Punkt im Code zu hinterlassen. Sie können mit `<!--` initiiert und mit `-->` abgeschlossen werden:

```
<!-- I'm an HTML comment! -->
```

Sie können in andere Inhalte integriert werden:

```
<h1>This part will be displayed <!-- while this will not be displayed -->.</h1>
```

Sie können auch mehrere Zeilen umfassen, um weitere Informationen bereitzustellen:

```
<!-- This is a multiline HTML comment.
  Whatever is in here will not be rendered by the browser.
  You can "comment out" entire sections of HTML code.
-->
```

Sie **können jedoch nicht** in einem anderen HTML-Tag wie dem folgenden angezeigt werden:

```
<h1 <!-- testAttribute="something" -->>This will not work</h1>
```

Dies erzeugt ungültige HTML als die gesamten `<h1 <!-- testAttribute="something" -->` Block ein einziges Start - Tag betrachtet würde `h1` mit einer anderen ungültigen darin enthaltenen Informationen, die von einer einzigen gefolgt `>` schließenden Klammer , die nichts tut.

Für die Kompatibilität mit Tools , die versuchen , HTML als XML oder SGML zu analysieren, den Körper Ihres Kommentars sollte zwei Striche nicht enthalten `--` .

Bedingte Kommentare für Internet Explorer

Bedingte Kommentare können verwendet werden, um den Code für verschiedene Versionen von Microsoft Internet Explorer anzupassen. Beispielsweise können verschiedene HTML-Klassen, Skript-Tags oder Stylesheets bereitgestellt werden. Bedingte Kommentare werden in den Internet Explorer-Versionen 5 bis 9 unterstützt. Ältere und neuere Internet Explorer-Versionen und alle Nicht-IE-Browser werden als "Downlevel" betrachtet und behandeln bedingte Kommentare als normale HTML-Kommentare.

Downlevel-versteckt

Downlevel-verborgene Kommentare funktionieren, indem der gesamte Inhalt in einen scheinbar normalen HTML-Kommentar gekapselt wird. Nur IE 5 bis 9 wird es immer noch als bedingten Kommentar lesen und den Inhalt entsprechend ausblenden oder anzeigen. In anderen Browsern wird der Inhalt ausgeblendet.

```
<!--[if IE]>
  Revealed in IE 5 through 9. Commented out and hidden in all other browsers.
<![endif]-->

<!--[if lt IE 8]>
  Revealed only in specified versions of IE 5-9 (here, IE less than 8).
<![endif]-->

<!--[if !IE]>
  Revealed in no browsers. Equivalent to a regular HTML comment.
<![endif]-->
```

```
<!--
  For purposes of comparison, this is a regular HTML comment.
-->
```

Downlevel-enthüllt

Diese unterscheiden sich geringfügig von versteckten Kommentaren, die nur in einem bestimmten Level enthalten sind: Nur der bedingte Kommentar selbst ist in der normalen Kommentarsyntax enthalten. Browser, die keine bedingten Kommentare unterstützen, werden sie einfach ignorieren und den Rest des Inhalts zwischen ihnen anzeigen.

```
<!--[if IE]>-->
  The HTML inside this comment is revealed in IE 5-9, and in all other browsers.
<!--<![endif]-->

<!--[if IE 9]>-->
  This is revealed in specified versions of IE 5-9, and in all other browsers.
<!--<![endif]-->

<!--[if !IE]>-->
  This is not revealed in IE 5-9. It's still revealed in other browsers.
<!--<![endif]-->
```

Leerzeichen zwischen Inline-Elementen auskommentieren

Inline-Anzeigeelemente, in der Regel " `span` oder " `a` , enthalten bis zu ein Leerzeichen vor und nach ihnen im Dokument. Um sehr lange Zeilen in der Markierung (die schwer lesbar sind) und unbeabsichtigte Leerräume (die sich auf die Formatierung auswirken) zu vermeiden, können die Leerräume auskommentiert werden.

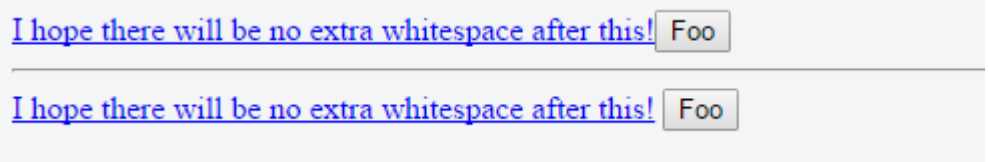
```
<!-- Use an HTML comment to nullify the newline character below: -->
<a href="#">I hope there will be no extra whitespace after this!</a><!--
--><button>Foo</button>
```

Versuchen Sie es ohne Kommentar zwischen den Inline-Elementen, und es gibt ein Leerzeichen zwischen ihnen. Manchmal ist das Abheben des Leerzeichens erwünscht.

Beispielcode:

```
<!-- Use an HTML comment to nullify the newline character below: -->
<a href="#">I hope there will be no extra whitespace after this!</a><!--
--><button>Foo</button>
<hr>
<!-- Without it, you can notice a small formatting difference: -->
<a href="#">I hope there will be no extra whitespace after this!</a>
<button>Foo</button>
```

Ausgabe:



Bemerkungen online lesen: https://riptutorial.com/de/html/topic/468/bemerkungen

Kapitel 8: Beschriftungselement

Syntax

- `<label>Example <input type="radio" name="r"></label>` // Ein Steuerelement umschließen
- `<label for="rad1">Example</label> <input id="rad1" type="radio" name="r">` // Verwendung `for` Attribut

Parameter

Attribute	Beschreibung
zum	Referenz auf das Ziel-ID-Element. Dh: `for="surname"`
bilden	**HTML5** , **[veraltet]** Verweis auf das Formular, das das Zielelement enthält. Labelelemente werden innerhalb eines `<form>` -Elements erwartet. Wenn `form="someFormId"` ist, können Sie das Label an einer beliebigen Stelle im Dokument platzieren.

Examples

Grundlegende Verwendung

Einfaches Formular mit Etiketten ...

```
<form action="/login" method="POST">

    <label for="username">Username:</label>
    <input id="username" type="text" name="username" />

    <label for="pass">Password:</label>
    <input id="pass" type="password" name="pass" />

    <input type="submit" name="submit" />

</form>
```

5

```
<form id="my-form" action="/login" method="POST">

    <input id="username" type="text" name="username" />

    <label for="pass">Password:</label>
    <input id="pass" type="password" name="pass" />

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

```
</form>

<label for="username" form="my-form">Username:</label>
```

Über das Label

Das `<label>` -Element wird verwendet, um auf ein Formularaktionselement zu verweisen.
In der **Benutzeroberfläche wird** es verwendet, um die Auswahl / Auswahl von Elementen wie Typ `radio` oder `checkbox` zu erleichtern.

`<label>` als Wrapper

Es kann das gewünschte Aktionselement einschließen

```
<label>
    <input type="checkbox" name="Cats">
    I like Cats!
</label>
```

(Klick auf den Text der `input` schaltet es Zustand / value)

`<label>` als Referenz

Wenn Sie das `for` Attribut verwenden, müssen Sie das Kontrollelement nicht als Abkömmling der `label` Der `for` Wert muss jedoch mit seiner ID übereinstimmen

```
<input id="cats" type="checkbox" name="Cats">
<label for="cats" >I like Cats!</label>
```

Hinweis
Verwenden Sie nicht mehr als ein Steuerelement in einem `<label>` -Element

Beschriftungselement online lesen: https://riptutorial.com/de/html/topic/1704/beschriftungselement

Kapitel 9: Bilder

Syntax

- `<img src="" alt="">`

Parameter

Parameter	Einzelheiten
`src`	Gibt die URL des Bildes an
`srcset`	Bilder zur Verwendung in verschiedenen Situationen (z. B. hochauflösende Displays, kleine Monitore usw.)
`sizes`	Bildgrößen zwischen Haltepunkten
`crossorigin`	Wie das Element Crossorigin-Anforderungen verarbeitet
`usemap`	Name der zu verwendenden Image-Map
`ismap`	Ob das Bild eine serverseitige Imagemap ist
`alt`	Alternativer Text, der angezeigt werden soll, wenn das Bild aus irgendeinem Grund nicht angezeigt werden konnte
`width`	Legt die Breite des Bildes fest (optional)
`height`	Legt die Höhe des Bildes fest (optional)

Examples

Bild erstellen

Verwenden Sie das Image-Tag, um ein Bild zu einer Seite hinzuzufügen.

Bild-Tags (`img`) haben keine schließenden Tags. Die zwei Hauptattribute, die Sie dem `img` Tag `src` , sind `src` , die Bildquelle und `alt` , ein alternativer Text, der das Bild beschreibt.

```
<img src="images/hello.png" alt="Hello World">
```

Sie können Bilder auch von einer Web-URL erhalten:

```
<img src="https://i.stack.imgur.com/ALgZi.jpg?s=48&g=1" alt="StackOverflow user Caleb
Kleveter">
```

Hinweis: Bilder werden technisch nicht in eine HTML-Seite eingefügt, Bilder werden mit HTML-Seiten verknüpft. Das `<img>` -Tag erstellt einen Speicherplatz für das referenzierte Bild.

Es ist auch möglich, Bilder mit base64 direkt in die Seite einzubetten:

```
<img src="..." alt="Hello World">
```

Tipp: Um ein Bild mit einem anderen Dokument zu verknüpfen, verschachteln Sie einfach das `<img>` -Tag in den `<a>` -Tags.

Bildbreite und -höhe

> **Hinweis:** Die **Breiten- und Höhenattribute sind für Bilder *nicht* veraltet** und waren es noch nie. Ihr Einsatz wurde jedoch viel strenger gemacht.

Die Abmessungen eines Bildes können mithilfe der Attribute `width` und `height` des Image-Tags festgelegt werden:

```
<img src="images/200x200-img.png" width="200" height="200" alt="A 200x200 image">
```

Durch die Angabe der `width` und `height` eines Bildes gibt Ihre Struktur dem Browser einen Hinweis, wie die Seite angeordnet werden soll, auch wenn Sie nur die tatsächliche Bildgröße angeben. Wenn die Bildmaße nicht angegeben werden, muss der Browser das Layout der Seite nach dem Laden des Bildes neu berechnen. Dies kann dazu führen, dass die Seite beim Laden "herumspringt".

4.1

Sie können dem Bild eine Breite und Höhe in der Anzahl der CSS-Pixel oder in Prozent der tatsächlichen Abmessungen des Bildes zuweisen.

Diese Beispiele sind alle gültig:

```
<img src="images/50x50-img.png" width="50" height="50" alt="A 50x50 image">
<img src="images/50x50-img.png" width="200" height="200" alt="A 200x200 image">
<img src="images/200x200-img.png" width="50" height="50" alt="A 50x50 image">
<img src="images/200x200-img.png" width="50%" height="50%" alt="A 100x100 image">
```

5

Die Breite und Höhe des Bildes muss in CSS-Pixeln angegeben werden. Ein Prozentwert ist kein gültiger Wert mehr. Wenn beide Attribute angegeben werden, müssen sie in eine der drei Formeln passen, die das Seitenverhältnis beibehalten. Obwohl gültig, sollten Sie die Breiten- und Höhenattribute nicht verwenden, um ein Bild auf eine größere Größe zu strecken.

Diese Beispiele sind gültig:

```
<img src="images/50x50-img.png" width="50" height="50" alt="A 50x50 image">
<img src="images/200x200-img.png" width="50" height="50" alt="A 50x50 image">
```

```
<img src="images/50x50-img.png" width="0" height="0" alt="A hidden tracking image">
```

Dieses Beispiel wird nicht empfohlen:

```
<img src="images/50x50-img.png" width="200" height="200" alt="A 200x200 image">
```

Diese Beispiele sind ungültig:

```
<img src="images/200x200-img.png" width="50%" height="50%" alt="A 100x100 image">
<img src="images/200x200-img.png" width="1" height="200" alt="A 1x200 image">
```

Wahl des alten Textes

Alttext wird von Bildschirmleseprogrammen für sehbehinderte Benutzer und von Suchmaschinen verwendet. Es ist daher wichtig, guten Text für Ihre Bilder zu schreiben.

Der Text sollte auch dann korrekt aussehen, wenn Sie das Bild durch das alt-Attribut ersetzen. Zum Beispiel:

```
<!-- Incorrect -->
<img src="anonymous.png" alt="Anonymous user avatar"/> An anonymous user wrote:
<blockquote>Lorem ipsum dolor sed.</blockquote>
<a href="https://google.com/"><img src="edit.png" alt="Edit icon"/></a> /
<a href="https://google.com/"><img src="delete.png" alt="Delete icon"/></a>
```

Ohne die Bilder würde dies folgendermaßen aussehen:

> Anonymer Benutzer avatar Ein anonymer Benutzer schrieb:
>
> > Lorem ipsum dolor sed.
>
> Symbol bearbeiten / Symbol löschen

Um dies zu korrigieren:

- Entfernen Sie den Alternativtext für den Avatar. Dieses Bild fügt Informationen für sehende Benutzer hinzu (ein leicht erkennbares Symbol, um anzuzeigen, dass der Benutzer anonym ist), diese Informationen sind jedoch bereits im Text verfügbar. [1]
- Entfernen Sie das "Symbol" aus dem Alt-Text für die Symbole. Zu wissen, dass dies ein Symbol wäre, wenn es dort wäre, hilft nicht, seinen tatsächlichen Zweck zu vermitteln.

```
<!-- Correct -->
<img src="anonymous.png" alt=""/> An anonymous user wrote:
<blockquote>Lorem ipsum dolor sed.</blockquote>
<a href="https://google.com/"><img src="edit.png" alt="Edit"/></a> /
<a href="https://google.com/"><img src="delete.png" alt="Delete"/></a>
```

> Ein anonymer Benutzer schrieb:
>
> > Lorem ipsum dolor sed.

Bearbeiten / Löschen

Fußnoten

[1] Es besteht ein semantischer Unterschied zwischen dem Einschließen eines leeren alt-Attributs und dem Ausschluss davon. Ein leeres alt-Attribut gibt an, dass das Bild *kein* wesentlicher Bestandteil des Inhalts ist (wie es in diesem Fall der Fall ist - es handelt sich lediglich um ein additives Bild, das zum Verständnis des Rests nicht erforderlich ist) und kann daher beim Rendern weggelassen werden. Jedoch zeigt das Fehlen eines alt - Attributs , dass das Bild ein wichtiger Teil des Inhalts *ist* , und dass es einfach kein Text äquivalent zum Rendern.

Responsives Bild mit dem Attribut srcset

Srcset mit Größen verwenden

```
<img sizes="(min-width: 1200px) 580px,
      (min-width: 640px) 48vw,
      98vw"
    srcset="img/hello-300.jpg 300w,
      img/hello-600.jpg 600w,
      img/hello-900.jpg 900w,
      img/hello-1200.jpg 1200w"
    src="img/hello-900.jpg" alt="hello">
```

`sizes` sind wie Medienabfragen. Sie beschreiben, wie viel Platz das Bild des Ansichtsfensters benötigt.

- Wenn der Darstellungsbereich größer als 1200px ist, ist das Bild genau 580px (z. B. ist unser Inhalt in einem Container zentriert, der maximal 1200px breit ist. Das Bild nimmt die Hälfte minus Ränder).
- Wenn der Darstellungsbereich zwischen 640px und 1200px liegt, nimmt das Bild 48% des Darstellungsbereichs ein (beispielsweise skaliert die Bildgröße auf unserer Seite und die Hälfte der Breite des Darstellungsbereichs minus Rändern).
- Wenn der Darstellungsbereich eine andere Größe hat, in unserem Fall weniger als 640 Pixel, nimmt das Bild 98% des Darstellungsbereichs ein (beispielsweise skaliert die Bildgröße bei unserer Seite und die gesamte Breite des Darstellungsbereichs minus Rändern). **Der Medienzustand muss für den letzten Artikel weggelassen werden.**

`srcset` sagt dem Browser nur, welche Bilder wir haben und welche Größe sie haben.

- `img/hello-300.jpg` ist 300px breit,
- `img/hello-600.jpg` ist 600px breit,
- `img/hello-900.jpg` ist 900px breit,
- `img/hello-1200.jpg` ist 1200px breit

`src` ist immer eine obligatorische Bildquelle. Bei Verwendung von `srcset` `src` ein Fallback-Bild, falls der Browser `srcset` nicht unterstützt.

Srcset ohne Größen verwenden

```
<img src="img/hello-300.jpg" alt="hello"
     srcset="img/hello-300.jpg 1x,
       img/hello-600.jpg 2x,
       img/hello-1200.jpg 3x">
```

`srcset` bietet eine Liste der verfügbaren Bilder mit dem Geräte-Pixel-Verhältnis `x` Deskriptor.

- Wenn das Geräte-Pixel-Verhältnis 1 ist, verwenden Sie `img/hello-300.jpg`
- Wenn das Geräte-Pixel-Verhältnis 2 ist, verwenden Sie `img/hello-600.jpg`
- Wenn das `img/hello-1200.jpg` 3 ist, verwenden Sie `img/hello-1200.jpg`

`src` ist immer eine obligatorische Bildquelle. Bei Verwendung von `srcset` `src` ein Fallback-Bild, falls der Browser `srcset` nicht unterstützt.

Responsives Bild mit Bildelement

Code

```
<picture>
  <source media="(min-width: 600px)" srcset="large_image.jpg">
  <source media="(min-width: 450px)" srcset="small_image.jpg">
  <img src="default_image.jpg" style="width:auto;">
</picture>
```

Verwendungszweck

Um verschiedene Bilder mit unterschiedlicher Bildschirmbreite anzuzeigen, müssen Sie alle Bilder, die das Quell-Tag verwenden, in ein Bild-Tag einfügen, wie im obigen Beispiel gezeigt.

Ergebnis

- Auf Bildschirmen mit einer Bildschirmbreite von> 600px wird large_image.jpg angezeigt
- Auf Bildschirmen mit einer Bildschirmbreite von> 450px wird small_image.jpg angezeigt
- Auf Bildschirmen mit anderer Bildschirmbreite wird default_image.jpg angezeigt

Bilder online lesen: https://riptutorial.com/de/html/topic/587/bilder

Kapitel 10: Charakter-Entitäten

Examples

Allgemeine Sonderzeichen

Einige Zeichen sind möglicherweise für HTML reserviert und können nicht direkt verwendet werden, da sie den tatsächlichen HTML-Code blockieren können. Wenn Sie beispielsweise versuchen, die linken und rechten spitzen Klammern (`<>`) im Quellcode anzuzeigen, kann dies zu unerwarteten Ergebnissen in der Ausgabe führen. In ähnlicher Weise werden im Quellcode geschriebene Leerzeichen möglicherweise nicht wie erwartet im Ausgabe-HTML-Code angezeigt. Einige, wie ☎, sind im ASCII-Zeichensatz nicht verfügbar.

Zu diesem Zweck werden Zeichenentitäten erstellt. Diese haben die Form `&entity_name;` oder `&entity_number;` . Im Folgenden sind einige der verfügbaren HTML-Entitäten aufgeführt.

Charakter	Beschreibung	Entitätsname	Nummer der Entität
""	schadensfreier Raum	` `	` `
"<"	weniger als	`<`	`&#60;`
">"	größer als	`>`	`&#62;`
"&"	Et-Zeichen	`&`	`&#38;`
"-"	EM Dash	`&mdash;`	`&#8212;`
"-"	en dash	`&ndash;`	`&#8211;`
"©"	Urheberrechte ©	`&copy;`	`&#169;`
"®"	eingetragene Marke	`&reg;`	`&#174;`
“™”	Warenzeichen	`&trade;`	`&#8482;`
"☎"	Telefon	`&phone;`	`&#9742;`

Also schreiben

© 2016 Stack Exchange Inc.

Der folgende HTML-Code wird verwendet:

```
<b>&copy; 2016 Stack Exchange Inc.</b>
```

Zeichenelemente in HTML

Bei der Entwicklung einer Webseite in HTML sind viele Symbole und Sonderzeichen erforderlich. Wie wir jedoch wissen, kann die direkte Verwendung von Zeichen den tatsächlichen HTML-Code beeinträchtigen, der bestimmte Zeichen reserviert und bestimmte Zeichen auf der Tastatur nicht verfügbar sind. Um den Konflikt zu vermeiden und gleichzeitig verschiedene Symbole in unserem Code verwenden zu können, stellt uns w3 org 'Character Entities' zur Verfügung.

Zeichenentitäten sind mit 'Entitätsname' vordefiniert - & entity_name; und 'Entity Number' - & entity_number; Daher müssen wir eines der beiden verwenden, damit das erforderliche Symbol auf unserer Seite dargestellt wird.

Die Liste der wenigen Zeichenelemente finden Sie unter https://dev.w3.org/html5/html-author/charref

Ein einfaches Beispiel mit der Verwendung der Zeichenentität für 'Lupe':

```
<input type="text" placeholder="  &#128269; Search"/>
```

was macht als

Charakter-Entitäten online lesen: https://riptutorial.com/de/html/topic/5229/charakter-entitaten

Kapitel 11: Computercode kennzeichnen

Syntax

- `<pre>Formatted text</pre>`
- `<code>Inline Code</code>`

Bemerkungen

Das `code` - Element sollte für jede Art von „Zeichenfolge, die ein Computer erkennen würde“ verwendet werden (HTML5), zum Beispiel:

- Quellcode
- Begriffe aus Markup- / Programmiersprachen (Elementnamen, Funktionsnamen usw.)
- Dateinamen

Verwandte Elemente

Für Variablen kann das Element `var` verwendet werden.

Für die Computerausgabe kann das `samp` Element verwendet werden.

Für Benutzereingaben kann das `kbd` Element verwendet werden.

Examples

Inline mit

Wenn ein Satz Computercode enthält (zum Beispiel der Name eines HTML - Elements) verwenden, das `code` - Element es zu markieren:

```
<p>The <code>a</code> element creates a hyperlink.</p>
```

Blockieren mit

```
 und
```

Wenn die Formatierung (weißer Raum, neue Linien, Vertiefung) der Code Angelegenheiten, verwenden Sie das `pre` - Element in Kombination mit dem `code` - Elemente:

```
<pre>
    <code>
    x = 42
    if x == 42:
        print "x is …           … 42"
    </code>
```

```
</pre>
```

Sie müssen noch Zeichen mit besonderer Bedeutung in HTML (wie `<` mit `<`) kennzeichnen. `<` also einen Block mit HTML-Code *anzeigen* (`<p>This is a paragraph.</p>`

```
<pre>
    <code>
    &lt;p>This is a paragraph.&lt;/p>
    </code>
</pre>
```

Computercode kennzeichnen online lesen:
https://riptutorial.com/de/html/topic/1836/computercode-kennzeichnen

Kapitel 12: Datenattribute

Syntax

- `<element data-custom-name="somevalue">`

Parameter

Wert	Beschreibung
irgendwas	Gibt den Wert des Attributs an (als Zeichenfolge).

Examples

Datenattribut verwenden

HTML5- `data-*` Attribute bieten eine bequeme Möglichkeit, Daten in HTML-Elementen zu speichern. Die gespeicherten Daten können mit JavaScript gelesen oder geändert werden

```
<div data-submitted="yes" class="user_profile">
  … some content …
</div>
```

- `data-*` ist `data-*` , dh der Name des `data-` kommt nach dem `data-` . Über diesen Namen kann auf das Attribut zugegriffen werden.
- Daten im String-Format (einschließlich `json`) können mit dem Attribut `data-*` gespeichert werden.

Ältere Browser unterstützen

Datenattribute wurden in HTML5 eingeführt, das von allen modernen Browsern unterstützt wird. Ältere Browser vor HTML5 erkennen die Datenattribute jedoch nicht.

In HTML-Spezifikationen müssen jedoch Attribute, die vom Browser nicht erkannt werden, in Ruhe gelassen werden. Der Browser ignoriert sie beim Rendern der Seite.

Webentwickler haben diese Tatsache genutzt, um Nicht-Standardattribute zu erstellen, bei denen es sich nicht um HTML-Spezifikationen handelt. Das `value` Attribut in der folgenden Zeile wird beispielsweise als nicht standardmäßiges Attribut betrachtet, da die Spezifikationen für das `<img>` -Tag kein `value` Attribut haben und es sich nicht um ein globales Attribut handelt:

```
<img src="sample.jpg" value="test" />
```

Dies bedeutet, dass `setAttribute` in älteren Browsern nicht unterstützt werden, sie jedoch

weiterhin funktionieren. Sie können sie mit den gleichen generischen JavaScript- `setAttribute` wie `getAttribute` und `getAttribute setAttribute` und `getAttribute` Sie können jedoch nicht die neue Eigenschaft `dataset` die nur in modernen Browsern unterstützt wird.

Datenattribute online lesen: https://riptutorial.com/de/html/topic/1182/datenattribute

Kapitel 13: Div Element

Einführung

Das div-Element in HTML ist ein Containerelement, das andere Elemente einkapselt und zum Gruppieren und Trennen von Teilen einer Webseite verwendet werden kann. Ein Div an sich repräsentiert nichts von Natur aus, sondern ist ein mächtiges Werkzeug im Webdesign. Dieses Thema behandelt den Zweck und die Anwendungen des div-Elements.

Syntax

- `<div>example div</div>`

Examples

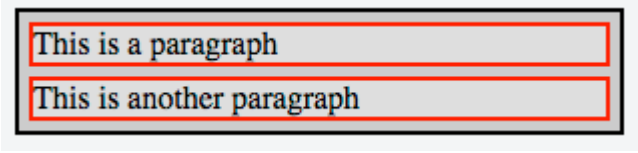
Verschachtelung

Es ist üblich, mehrere `<div>` in einem anderen `<div>` . Dies wird normalerweise als "Verschachtelungs" -Elemente bezeichnet und ermöglicht die weitere Unterteilung der Elemente in Unterabschnitte oder die Entwicklung von Entwicklern mit CSS-Stil.

Das `<div class="outer-div">` wird verwendet, um zwei `<div class="inner-div">` -Elemente zusammenzufassen. jedes enthält ein `<p>` -Element.

```
<div class="outer-div">
  <div class="inner-div">
    <p>This is a paragraph</p>
  </div>
  <div class="inner-div">
    <p>This is another paragraph</p>
  </div>
</div>
```

Dies führt zu folgendem Ergebnis (CSS-Stile werden aus Gründen der Übersichtlichkeit angewendet):



Inline- und Blockelemente verschachteln Beim Verschachteln von Elementen sollten Sie berücksichtigen, dass es Inline- und Blockelemente gibt. Während Blockelemente "einen Zeilenumbruch im Hintergrund hinzufügen", dh andere verschachtelte Elemente werden automatisch in der nächsten Zeile angezeigt, können Inline-Elemente standardmäßig nebeneinander positioniert werden

Vermeiden Sie tiefe `<div>` -Verschachtelung

Ein tiefes und häufig verwendetes verschachteltes Containerlayout zeigt einen schlechten Codierstil.

Abgerundete Ecken oder ähnliche Funktionen erzeugen häufig einen solchen HTML-Code. Für die meisten Browser der letzten Generation gibt es CSS3-Gegenstücke. Verwenden Sie möglichst wenig HTML-Elemente, um das Verhältnis von Inhalten zu Tags zu erhöhen und die Seitenlast zu reduzieren, was zu einem besseren Ranking in Suchmaschinen führt.

`div` section Element sollte nicht tiefer als 6 Ebenen verschachtelt sein.

Grundlegende Verwendung

Das `<div>` -Element hat normalerweise keine spezifische semantische Bedeutung, sondern stellt lediglich eine Division dar und wird normalerweise zum Gruppieren und Einkapseln anderer Elemente in einem HTML-Dokument und zum Trennen dieser Elemente von anderen Inhaltsgruppen verwendet. Daher wird jeder `<div>` am besten durch seinen Inhalt beschrieben.

```
<div>
  <p>Hello! This is a paragraph.</p>
</div>
```

Das `div` Element ist normalerweise ein Element auf Blockebene. Dies bedeutet, dass es einen Block eines HTML-Dokuments trennt und die maximale Breite der Seite einnimmt. Browser haben normalerweise die folgende Standard-CSS-Regel:

```
div {
  display: block;
}
```

> **Das World Wide Web Consortium (W3C)** empfiehlt dringend, das div-Element als letzten Ausweg anzusehen, wenn kein anderes Element geeignet ist. Die Verwendung geeigneter Elemente anstelle des div-Elements führt zu einer besseren Zugänglichkeit für Leser und zu einer einfacheren Wartung für Autoren.

Ein Blogbeitrag würde beispielsweise mit `<article>` , einem Kapitel mit `<section>` , den Navigationshilfen einer Seite mit `<nav>` und einer Gruppe von Formularsteuerelementen mit `<fieldset>` .

div-Elemente können aus stilistischen Gründen nützlich sein oder mehrere Absätze innerhalb eines Abschnitts umschließen, die alle auf ähnliche Weise kommentiert werden sollen.

Div Element online lesen: https://riptutorial.com/de/html/topic/1468/div-element

Kapitel 14: Doktypen

Einführung

Doctypes - kurz für 'Dokumenttyp' - helfen Browsern dabei, die HTML-Version zu verstehen, in der das Dokument geschrieben ist, um die Interpretierbarkeit zu verbessern. Doctype-Deklarationen sind keine HTML-Tags und gehören ganz oben in ein Dokument. In diesem Thema werden die Struktur und Deklaration verschiedener Doktypen in HTML erläutert.

Syntax

- <! DOCTYPE [versionsspezifische Zeichenfolge]>

Bemerkungen

Die Deklaration `<!DOCTYPE>` ist kein HTML-Tag. Es wird verwendet, um anzugeben, welche HTML-Version das Dokument verwendet. Dies wird als Dokumenttypdeklaration (DTD) bezeichnet.

Die Deklaration `<!DOCTYPE>` NICHT zwischen Groß- und Kleinschreibung. Um zu überprüfen, ob der HTML-Code Ihrer Webseiten gültig ist, besuchen Sie den Validierungsdienst von W3C .

- Einige alte Versionen von IE unterstützen einige HTML-Tags nur, wenn ein ordnungsgemäßer Doctype verfügbar ist.
- Es ist *wichtig,* dass ein Doctype als deklariert wird, um sicherzustellen, dass der Browser den Quirksmodus nicht verwendet. Weitere Informationen zu MDN.

Examples

Doctype hinzufügen

Die Deklaration `<!DOCTYPE>` sollte vor dem `<html>` -Tag immer oben im HTML-Dokument stehen.

5

Siehe HTML 5 Doctype für Details auf den HTML - 5 Doctype.

```
<!DOCTYPE html>
```

4.01

Informationen dazu, wie sich diese Typen unterscheiden, finden Sie unter HTML 4.01-Doktypen .

Streng

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

Übergang

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Frameset

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

HTML 4.01 Doctypes

Die HTML 4.01-Spezifikation stellt verschiedene Arten von Doktypen bereit, mit denen verschiedene Arten von Elementen innerhalb des Dokuments angegeben werden können.

HTML 4.01 Streng

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

Umfasst alle HTML-Elemente und -Attribute, jedoch **keine Präsentationselemente oder veraltete Elemente. Framesets sind nicht zulässig** .

HTML 4.01 Übergang

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Umfasst alle HTML-Elemente und Attribute sowie Präsentations- und veraltete Elemente. **Framesets sind jedoch nicht zulässig** .

HTML 4.01 Frameset

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

Umfasst alle HTML-Elemente und -Attribute sowie Präsentations- und veraltete Elemente. Framesets sind erlaubt.

HTML 5 Doctype

HTML5 basiert nicht auf SGML und erfordert daher keinen Verweis auf eine DTD.

HTML 5 Doctype-Deklaration:

```
<!DOCTYPE html>
```

Fallunempfindlichkeit

Gemäß der W3.org HTML 5 `DOCTYPE` :

> Ein DOCTYPE muss in dieser Reihenfolge aus den folgenden Komponenten bestehen:
>
> 1. Eine Zeichenfolge, **bei der** die Zeichenfolge `"<!DOCTYPE"` für die Zeichenfolge `"<!DOCTYPE"`

Daher gelten auch folgende `DOCTYPE` :

```
<!doctype html>
<!dOCtyPe html>
<!DocTYpe html>
```

In diesem SO-Artikel wird das Thema ausführlich behandelt: Doctype in Groß- oder Kleinschreibung?

Alte Lehren

HTML 3.2

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
```

HTML 3.2 wird von den meisten verwendeten Browsern gut unterstützt. HTML 3.2 bietet jedoch nur eine begrenzte Unterstützung für Stylesheets und keine Unterstützung für HTML 4-Funktionen wie Frames und Internationalisierung.

HTML 2.0

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
```

HTML 2.0 wird von Browsern weitgehend unterstützt, es gibt jedoch keine Unterstützung für Tabellen, Frames und Internationalisierung sowie viele häufig verwendete Präsentationselemente und -attribute.

Doktypen online lesen: https://riptutorial.com/de/html/topic/806/doktypen

Kapitel 15: Einbetten

Parameter

Parameter	Einzelheiten
`src`	Adresse der Ressource
`type`	Art der eingebetteten Ressource
`width`	Horizontale Abmessung
`height`	Vertikale Abmessung

Examples

Grundlegende Verwendung

Das `embed` Tag ist neu in HTML5. Dieses Element stellt einen Integrationspunkt für eine externe Anwendung (normalerweise kein HTML) oder interaktiven Inhalt bereit.

```
<embed src="myflash.swf">
```

MIME-Typ definieren

Der MIME- Typ muss mit dem `type` Attribut definiert werden.

```
<embed type="video/mp4" src="video.mp4" width="640" height="480">
```

Einbetten online lesen: https://riptutorial.com/de/html/topic/8123/einbetten

Kapitel 16: Eingabesteuerelemente

Einführung

Eingabetags, eine Schlüsselkomponente interaktiver Websysteme, sind HTML-Elemente, die eine bestimmte Form der Eingabe von Benutzern annehmen. Verschiedene Arten von Eingabeelementen können die eingegebenen Daten so regeln, dass sie in ein bestimmtes Format passen und die Passworteingabe schützen.

Syntax

- `<input type="" name="" value="">`

Parameter

Parameter	Einzelheiten
Klasse	Gibt die Klasse der Eingabe an
Ich würde	Gibt die ID der Eingabe an
Art	Gibt den Typ der Eingabesteuerung an, die angezeigt werden soll. Zulässige Werte sind `hidden` , `text` , `tel` , `url` , `email` , `password` , `date` , `time` , `number` , `range` , `color` , `checkbox` , `radio` , `file` , `submit` , `image` , `reset` und `button` . Der Standardwert ist `text` wenn er nicht angegeben wurde, der Wert ungültig ist oder der Browser den angegebenen Typ nicht unterstützt.
Name	Gibt den Namen der Eingabe an
deaktiviert	Boolescher Wert, der angibt, dass die Eingabe deaktiviert werden soll. Deaktivierte Steuerelemente können nicht bearbeitet werden, werden beim Senden des Formulars nicht gesendet und erhalten keinen Fokus.
geprüft	Wenn der Wert des type-Attributs radio oder ein Kontrollkästchen ist, zeigt das Vorhandensein dieses booleschen Attributs an, dass das Steuerelement standardmäßig ausgewählt ist. ansonsten wird es ignoriert.
mehrere	**HTML5** Gibt an, dass mehrere Dateien oder Werte übergeben werden können (Gilt nur für `file` und `email` Eingaben).
Platzhalter	**HTML5** Ein Hinweis an den Benutzer, was im Steuerelement eingegeben werden kann. Der Platzhaltertext darf keine

Parameter	Einzelheiten
	Zeilenumbrüche oder Zeilenvorschübe enthalten
Autovervollständigung	**HTML5** Gibt an, ob der Wert des Steuerelements vom Browser automatisch ausgefüllt werden kann.
schreibgeschützt	Boolescher Wert, der angibt, dass die Eingabe nicht bearbeitet werden kann. Readonly-Steuerelemente werden immer noch bei der Formularübermittlung gesendet, erhalten jedoch keinen Fokus. **HTML5:** Dieses Attribut wird ignoriert, wenn der Wert des Attributs `type` entweder auf `hidden` , `range` , `color` , `checkbox` , `radio` , `file` oder `button` .
erforderlich	**HTML5** Gibt an, dass ein Wert vorhanden sein muss, oder das Element muss überprüft werden, damit das Formular gesendet werden kann
alt	Ein alternativer Text für Bilder, falls diese nicht angezeigt werden.
Autofokus	Das `<input>` -Element sollte beim Laden der Seite den Fokus erhalten.
Wert	Gibt den Wert des Elements `<input>` .
Schritt	Das `step` gibt die gesetzlichen Nummernintervalle an. Es funktioniert mit den folgenden Eingabetypen: `number` , `range` , `date` , `datetime-local` , `month` , `time` und `week` .

Bemerkungen

Wie bei anderen void HTML5-Elementen ist `<input>` selbstschließend und kann `<input />` . HTML5 erfordert diesen Schrägstrich nicht.

Folgendes sind gültige Eingabetypen in HTML:

- `button`
- `checkbox`
- `file`
- `hidden`
- `image`
- `password`
- `radio`
- `reset`
- `submit`
- `text` (Standardwert)

5

Das Folgende sind neu eingeführte Eingabetypen als Teil des HTML 5-Standards. Einige dieser

Typen werden nicht von allen Webbrowsern unterstützt. Wenn ein Typ nicht unterstützt wird, verwendet das Eingabeelement standardmäßig den `text` .

- `color`
- `date`
- `datetime` (veraltet und veraltet)
- `datetime-local`
- `email`
- `month`
- `number`
- `range`
- `search`
- `tel`
- `time`
- `url`
- `week`

Um zu prüfen, welche Browser welche Typen unterstützen, können Sie zu caniuse.com gehen.

Examples

Kontrollkästchen und Optionsfelder

Überblick

Kontrollkästchen und Optionsfelder werden mit dem HTML-Tag `<input>` und ihr Verhalten wird in der HTML-Spezifikation definiert .

Das einfachste Kontrollkästchen oder Optionsfeld ist ein `<input>` Element mit einem `type` - Attribute `checkbox` oder `radio` jeweils:

```
<input type="checkbox">
<input type="radio">
```

Ein einzelnes eigenständiges Kontrollkästchenelement wird für eine einzelne binäre Option verwendet, beispielsweise für eine Ja-Nein-Frage. Ankreuzfelder sind unabhängig, dh der Benutzer kann in einer Gruppe von Ankreuzfeldern beliebig viele Auswahlmöglichkeiten auswählen. Wenn Sie also ein Kontrollkästchen aktivieren, werden die anderen Kontrollkästchen in der Kontrollkästchengruppe *nicht* deaktiviert.

Radio - Buttons in der Regel in Gruppen kommen identifiziert (wenn es nicht mit einem anderen Optionsfeld gruppiert ist, werden Sie wahrscheinlich eine Checkbox stattdessen verwenden gemeint ist) unter Verwendung des gleichen `name` - Attribut für alle Tasten in dieser Gruppe. Die Auswahl der Optionsfelder schließt sich *gegenseitig aus* , dh der Benutzer kann nur eine Auswahl aus einer Gruppe von Optionsfeldern auswählen. Wenn ein Optionsfeld aktiviert ist, wird jedes andere Optionsfeld mit demselben `name` , das zuvor aktiviert wurde, deaktiviert.

Beispiel:

```
<input type="radio" name="color" id="red" value="#F00">
<input type="radio" name="color" id="green" value="#0F0">
<input type="radio" name="color" id="blue" value="#00F">
```

Bei der Anzeige werden Optionsfelder als Kreis (nicht markiert) oder als gefüllter Kreis (aktiviert) angezeigt. Kontrollkästchen werden als Quadrat (nicht angehakt) oder als ausgefülltes Quadrat (aktiviert) angezeigt. Je nach Browser und Betriebssystem hat das Quadrat manchmal abgerundete Ecken.

Attribute

Kontrollkästchen und Optionsfelder verfügen über eine Reihe von Attributen, um ihr Verhalten zu steuern:

`value`

Wie jedes andere Eingabeelement gibt das `value` Attribut den Zeichenfolgenwert an, der der Schaltfläche im Falle der Formularübermittlung zugeordnet werden soll. Allerdings sind Checkboxen und Radiobuttons spezielle, dass , wenn der Wert nicht angegeben wird, wird standardmäßig `on` , wenn vorgelegt, anstatt einen leeren Wert zu senden. Das `value` Attribut wird nicht im Erscheinungsbild der Schaltfläche angezeigt.

`checked`

Das `checked` Attribut gibt den Anfangsstatus eines Kontrollkästchens oder eines Optionsfelds an. Dies ist ein boolesches Attribut und kann weggelassen werden.

Jede dieser Optionen ist eine gültige, gleichwertige Methode zum Definieren eines aktivierten Optionsfelds:

```
<input checked>
<input checked="">
<input checked="checked">
<input checked="ChEcKeD">
```

Das Fehlen des `checked` Attributs ist die einzige gültige Syntax für eine ungeprüfte Schaltfläche:

```
<input type="radio">
<input type="checkbox">
```

Wenn Sie ein `<form>` zurücksetzen, werden die Kontrollkästchen und Optionsfelder auf den Status ihres `checked` Attributs zurückgesetzt.

Zugänglichkeit

Etiketten

Um den Schaltflächen einen Kontext zu geben und den Benutzern zu zeigen, wozu jede Schaltfläche dient, sollte jede von ihnen eine Bezeichnung haben. Dies kann mit einem `<label>` -Element durchgeführt werden, um die Schaltfläche einzuwickeln. Dadurch wird das Etikett auch anklickbar, sodass Sie die entsprechende Schaltfläche auswählen.

Beispiel:

```
<label>
  <input type="radio" name="color" value="#F00">
  Red
</label>
```

oder mit einem `<label>` -Element mit einem `for` Attribut auf das `id` Attribut der Schaltfläche:

```
<input type="checkbox" name="color" value="#F00" id="red">
<label for="red">Red</label>
```

Schaltflächengruppen

Da jedes Optionsfeld die anderen in der Gruppe beeinflusst, ist es üblich, eine Beschriftung oder einen Kontext für die gesamte Gruppe von Optionsfeldern anzugeben.

Um eine Beschriftung für die gesamte Gruppe bereitzustellen, sollten die Optionsfelder in einem `<fieldset>` -Element mit einem `<legend>` -Element enthalten sein.

Beispiel:

```
<fieldset>
  <legend>Theme color:</legend>
  <p>
    <input type="radio" name="color" id="red" value="#F00">
    <label for="red">Red</label>
  </p>
  <p>
    <input type="radio" name="color" id="green" value="#0F0">
    <label for="green">Green</label>
  </p>
  <p>
    <input type="radio" name="color" id="blue" value="#00F">
    <label for="blue">Blue</label>
  </p>
</fieldset>
```

Kontrollkästchen können auch auf ähnliche Weise gruppiert werden, wobei ein Feldset und eine Legende die Gruppe der zugehörigen Kontrollkästchen angeben. Beachten Sie jedoch, dass Kontrollkästchen *nicht* denselben Namen haben sollten, da sie sich nicht gegenseitig ausschließen. Dies führt dazu, dass das Formular mehrere Werte für denselben Schlüssel übermittelt, und nicht alle serverseitigen Sprachen behandeln dies auf dieselbe Weise (undefiniertes Verhalten). Jedes Kontrollkästchen sollte entweder einen eindeutigen Namen haben oder eine Reihe von eckigen Klammern (`[]`) verwenden, um anzugeben, dass das Formular ein Array mit Werten für diesen Schlüssel übermitteln soll. Welche Methode Sie wählen, sollte davon

abhängen, wie Sie die Formulardaten auf Client- oder Serverseite behandeln. Sie sollten auch die Legende kurz halten, da einige Kombinationen von Browsern und Bildschirmlesern die Legende vor jedem Eingabefeld im Feldset lesen.

Versteckt

```
<input type="hidden" name="inputName" value="inputValue">
```

Eine verborgene Eingabe ist für den Benutzer nicht sichtbar, der Wert wird jedoch an den Server gesendet, wenn das Formular dennoch gesendet wird.

Passwort

```
<input type="password" name="password">
```

Das Eingabeelement mit einem Typattribut, dessen Wert " `password` erstellt ein einzeiliges Textfeld ähnlich dem Eingabetyp `type=text` . Der Text wird jedoch nicht angezeigt, wenn er vom Benutzer eingegeben wird.

```
<input type="password" name="password" placeholder="Password">
```

Platzhaltertext wird im Klartext angezeigt und automatisch überschrieben, wenn ein Benutzer mit der Eingabe beginnt.

Hinweis: Einige Browser und Systeme ändern das Standardverhalten des Kennwortfelds, um auch das zuletzt eingegebene Zeichen für eine kurze Zeit anzuzeigen.

einreichen

```
<input type="submit" value="Submit">
```

Bei der Übermittlungseingabe wird eine Schaltfläche erstellt, die beim Klicken das darin enthaltene Formular übermittelt.

Sie können das `<button>` -Element auch verwenden, wenn Sie eine Senden-Schaltfläche benötigen, die einfacher gestaltet werden kann oder andere Elemente enthält:

```
<button type="submit">
  <img src="submit-icon.jpg" /> Submit
</button>
```

Datei

```
<input type="file" name="fileSubmission">
```

Dateieingaben ermöglichen Benutzern die Auswahl einer Datei aus ihrem lokalen Dateisystem zur Verwendung mit der aktuellen Seite. Wenn sie in Verbindung mit einem `form` verwendet werden, können sie Benutzern das Hochladen von Dateien auf einen Server ermöglichen (weitere Informationen finden Sie unter Hochladen von Dateien).

Im folgenden Beispiel können Benutzer die `file` verwenden, um eine Datei aus ihrem Dateisystem auszuwählen und diese Datei in ein Skript auf dem Server mit dem Namen `upload_file.php` .

```
<form action="upload_file.php" method="post" enctype="multipart/form-data">
    Select file to upload:
    <input type="file" name="fileSubmission" id="fileSubmission">
    <input type="submit" value="Upload your file" name="submit">
</form>
```

Mehrere Dateien

Durch das Hinzufügen des `multiple` kann der Benutzer **mehrere** Dateien auswählen:

```
<input type="file" name="fileSubmission" id="fileSubmission" multiple>
```

Dateien annehmen

Accept-Attribut gibt die Dateitypen an, die der Benutzer auswählen kann. ZB `.png` , `.gif` , `.jpeg` .

```
<input type="file" name="fileSubmission" accept="image/x-png,image/gif,image/jpeg" />
```

Eingabevalidierung

Die Validierung der HTML-Eingabe wird automatisch vom Browser durchgeführt, basierend auf speziellen Attributen des Eingabeelements. Es könnte die JavaScript-Eingabevalidierung teilweise oder vollständig ersetzen. Diese Art der Validierung kann vom Benutzer über speziell gestaltete HTTP-Anforderungen umgangen werden, so dass die serverseitige Eingabevalidierung nicht ersetzt wird. Die Validierung erfolgt nur beim Versuch, das Formular zu senden. Daher müssen sich alle eingeschränkten Eingaben innerhalb eines Formulars befinden, damit die Validierung erfolgen kann (sofern Sie nicht JavaScript verwenden). Beachten Sie, dass deaktivierte oder schreibgeschützte Eingänge keine Validierung auslösen.

Einige neuere Eingabetypen (wie `email` , `url` , `tel` , `date` und viele andere) werden automatisch validiert und erfordern keine eigenen Validierungseinschränkungen.

5

Erforderlich

Verwenden Sie das `required` Attribut, um anzugeben, dass ein Feld ausgefüllt werden muss, um die Validierung zu bestehen.

```
<input required>
```

Mindest- / Höchstlänge

Verwenden Sie die Attribute `minlength` und `maxlength` , um die Längenanforderungen anzugeben. Die meisten Browser verhindern, dass der Benutzer mehr als *maximal* Zeichen in das Feld eingibt. Dadurch wird verhindert, dass der Eintrag ungültig wird, noch bevor er die Übermittlung versucht.

```
<input minlength="3">
<input maxlength="15">
<input minlength="3" maxlength="15">
```

Bereich angeben

Verwenden `min` und `max` - Attribut des Zahlenbereich um einen Benutzer zu beschränken Eingang in einen Eingang vom Typ `number` oder `range`

```
Marks: <input type="number" size="6" name="marks" min="0" max="100" />
Subject Feedback: <input type="range" size="2" name="feedback" min="1" max="5" />
```

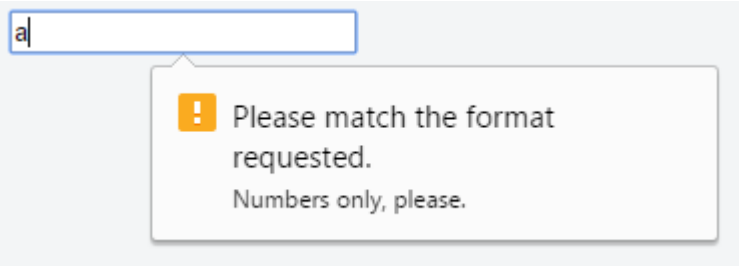
5

Übereinstimmung mit einem Muster

Verwenden Sie für mehr Kontrolle das `pattern` , um einen regulären Ausdruck anzugeben, der abgeglichen werden muss, um die Validierung zu bestehen. Sie können auch einen `title` angeben, der in der Bestätigungsnachricht enthalten ist, wenn das Feld nicht besteht.

```
<input pattern="\d*" title="Numbers only, please.">
```

In Google Chrome Version 51 wird die Nachricht angezeigt, wenn Sie versuchen, das Formular mit einem ungültigen Wert in diesem Feld zu senden:



Nicht alle Browser zeigen eine Meldung für ungültige Muster an, obwohl die meisten gängigen modernen Browser voll unterstützt werden.

Überprüfen Sie die aktuelle Unterstützung von CanIUse und implementieren Sie sie entsprechend.

5

Dateityp akzeptieren

Für Eingabefelder vom Typ `file` können nur bestimmte Dateitypen akzeptiert werden, z. B. Videos, Bilder, Audios, bestimmte Dateierweiterungen oder bestimmte Medientypen . Zum Beispiel:

```
<input type="file" accept="image/*" title="Only images are allowed">
```

Mehrere Werte können mit einem Komma angegeben werden, zB:

```
<input type="file" accept="image/*,.rar,application/zip">
```

Hinweis: Durch `novalidate` Hinzufügen `novalidate` Attributs `novalidate` zum `form` oder `formnovalidate` Attribut zur Schaltfläche " `formnovalidate` wird die Überprüfung von Formularelementen verhindert. Zum Beispiel:

```
<form>
    <input type="text" name="name" required>
    <input type="email" name="email" required>
    <input pattern="\d*" name="number" required>

    <input type="submit" value="Publish"> <!-- form will be validated -->
    <input type="submit" value="Save" formnovalidate> <!-- form will NOT be validated -->
</form>
```

Das Formular enthält Felder, die zum "Veröffentlichen" des Entwurfs erforderlich sind, jedoch nicht zum "Speichern" des Entwurfs.

Zurücksetzen

```
<input type="reset" value="Reset">
```

Durch die Eingabe des Typs `reset` wird eine Schaltfläche erstellt, durch die alle Eingaben in der Form, in der sie enthalten sind, auf ihren Standardzustand zurückgesetzt werden.

- Text in einem Eingabefeld wird auf leer oder seinem Standardwert zurückgesetzt werden (unter Verwendung des angegebenen `value` Attribute).
- Alle Optionen in einem Auswahlmenü werden deaktiviert, sofern sie nicht das `selected` Attribut haben.
- Alle Kontrollkästchen und Optionsfelder werden abgewählt werden , wenn sie die haben `checked` Attribut.

Hinweis: Eine Reset-Schaltfläche muss sich innerhalb eines `<form>` -Elements befinden oder mit diesem verbunden sein (über das `form`), um Auswirkungen zu haben. Die Schaltfläche setzt nur die Elemente innerhalb dieses Formulars zurück.

Nummer

5

```
<input type="number" value="0" name="quantity">
```

Das Eingabeelement mit einem Typattribut, dessen Wert " `number` stellt ein genaues Steuerelement dar, um den Wert des Elements auf eine Zeichenfolge festzulegen, die eine Zahl darstellt.

Bitte beachten Sie, dass dieses Feld keine korrekte Nummer garantiert. Es erlaubt nur alle Symbole, die in einer beliebigen reellen Zahl verwendet werden können. Der Benutzer kann beispielsweise einen Wert wie `e1e-,0` .

Tel

```
<input type="tel" value="+8400000000">
```

Das Eingabeelement mit einem Typattribut, dessen Wert `tel` repräsentiert ein einzeiliges Klartext-Bearbeitungssteuerelement zum Eingeben einer Telefonnummer.

Email

5

Das `<input type="email">` wird für Eingabefelder verwendet, die eine E-Mail-Adresse enthalten sollen.

```
<form>
  <label>E-mail: <label>
  <input type="email" name="email">
</form>
```

Die E-Mail-Adresse kann bei der Übermittlung automatisch überprüft werden, abhängig von der Browserunterstützung.

Taste

```
<input type="button" value="Button Text">
```

Mit Schaltflächen können Aktionen ausgelöst werden, ohne dass das Formular gesendet wird. Sie können das Element `<button>` verwenden, wenn Sie eine Schaltfläche benötigen, die einfacher gestaltet werden kann oder andere Elemente enthält:

```
<button type="button">Button Text</button>
```

Schaltflächen werden normalerweise bei einem "Onclick" -Ereignis verwendet:

```
<input type="button" onclick="alert('hello world!')" value="Click Me">
```

oder

```
<button type="button" onclick="alert('hello world!')">Click Me</button>
```

Attribute

[name]

Der `name` der Schaltfläche, die mit den Formulardaten übermittelt wird.

[type]

Die `type` der Schaltfläche.

Mögliche Werte sind:

`submit` : Über die Schaltfläche werden die Formulardaten an den Server `submit` . Dies ist die Standardeinstellung, wenn das Attribut nicht angegeben ist oder wenn das Attribut dynamisch in einen leeren oder ungültigen Wert geändert wird.

`reset` : Die Schaltfläche setzt alle Steuerelemente auf ihre Ausgangswerte zurück.

`button` : Die Schaltfläche hat kein Standardverhalten. Mit den Ereignissen des Elements können clientseitige Skripts verknüpft sein, die beim Auftreten der Ereignisse ausgelöst werden.

`menu` : Die Schaltfläche öffnet ein Popup-Menü, das über das angegebene Element definiert wird.

[value]

Der Anfangswert der Schaltfläche.

5

Zusätzliche Attribute für Senden-Schaltflächen

Attribut	Beschreibung
`form`	Gibt die ID des Formulars an, zu dem die Schaltfläche gehört. Wenn keines angegeben ist, gehört es zu seinem Vorfahren-Formularelement (sofern vorhanden).
`formaction`	Gibt an, wohin die Formulardaten gesendet werden sollen wenn das Formular mit dieser Schaltfläche gesendet wird.
`formenctype`	Gibt an, wie die Formulardaten codiert werden sollen beim Senden an den Server über diese Schaltfläche. Kann nur mit `formmethod="post"` .

Attribut	Beschreibung
`formmethod`	Gibt die zu verwendende HTTP-Methode an (POST oder GET). beim Senden von Formulardaten über diese Schaltfläche.
`formnovalidate`	Gibt an, dass die Formulardaten beim Senden nicht überprüft werden sollen.
`formtarget`	Gibt an, wo die empfangene Antwort angezeigt werden soll nach dem Absenden des Formulars mit dieser Schaltfläche.

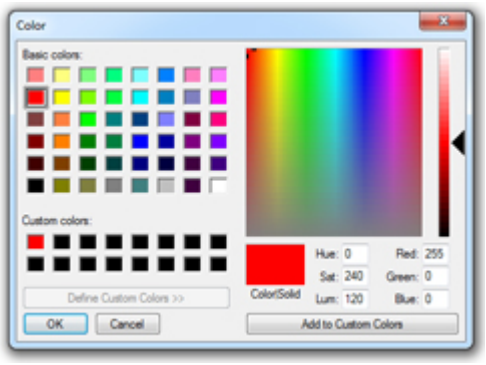
Farbe

5

```
<input type="color" name="favcolor" value="#ff0000">
```

In unterstützten Browsern erstellt das Eingabeelement mit einem Typattribut, dessen Wert `color` ist, ein schaltflächenförmiges Steuerelement mit einer Farbe, die dem Wert des Attributs `color` (Standardwert schwarz, wenn value nicht angegeben ist oder ein ungültiges Hexadezimalformat ist).

Durch Klicken auf diese Schaltfläche wird das Farb-Widget des Betriebssystems geöffnet, in dem der Benutzer eine Farbe auswählen kann.



Fallback für Browser, die diesen Eingabetyp nicht unterstützen, ist ein regulärer Eingabetyp `type=text` .

URL

5

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

Dies wird für Eingabefelder verwendet, die eine URL-Adresse enthalten sollen.

Abhängig von der Browserunterstützung kann das `url` Feld beim `url` automatisch überprüft werden.

Einige Smartphones erkennen den `url` Typ und fügen der Tastatur ".com" hinzu, um der URL-Eingabe zu entsprechen.

Datum

5

```
<input type="date" />
```

Auf dem Bildschirm wird eine Datumsauswahl angezeigt, aus der Sie ein Datum auswählen können. Dies wird in Firefox oder Internet Explorer nicht unterstützt.

DateTime-Local

5

```
<input type="datetime-local" />
```

Abhängig von der Browserunterstützung wird eine Datums- und Uhrzeitauswahl eingeblendet, auf der Sie ein Datum und eine Uhrzeit auswählen können.

Bild

```
<input type="image" src="img.png" alt="image_name" height="50px" width="50px"/>
```

Ein Bild. Sie müssen das Attribut src verwenden, um die Quelle des Bildes zu definieren, und das Attribut alt, um alternativen Text zu definieren. Sie können die Attribute height und width verwenden, um die Größe des Bildes in Pixel zu definieren.

Angebot

5

```
<input type="range" min="" max="" step="" />
```

Ein Steuerelement zur Eingabe einer Zahl, deren genauer Wert nicht wichtig ist.

Attribut	Beschreibung	Standardwert
Mindest	Mindestwert für den Bereich	0
max	Maximalwert für den Bereich	100
Schritt	Betrag, der in jedem Inkrement um jeweils mehr erhöht wird.	1

Monat

5

```
<input type="month" />
```

Abhängig von der Browserunterstützung zeigt ein Steuerelement den Monat an.

Zeit

5

```
<input type="time" />
```

Der `time` markiert dieses Element als String Annahme einer Zeit darstellt. Das Format ist in RFC 3339 definiert und sollte eine Teilzeit sein, wie z

```
19:04:39
08:20:39.04
```

Derzeit unterstützen alle Versionen von Edge, Chrome, Opera und Chrome für Android den Typ "time". Die neueren Versionen des Android-Browsers, insbesondere 4.4 und höher, unterstützen dies. Safari für iOS bietet teilweise Unterstützung, unterstützt jedoch nicht die Attribute min, max und step.

Woche

5

```
<input type="week" />
```

Abhängig von der Browserunterstützung wird ein Steuerelement zum Eingeben einer Wochennummer und einer Wochennummer ohne Zeitzone angezeigt.

Text

Der einfachste Eingabetyp und die Standardeingabe, wenn kein `type` angegeben ist. Dieser Eingabetyp definiert ein einzeiliges Textfeld, bei dem Zeilenumbrüche automatisch aus dem Eingabewert entfernt werden. Alle anderen Zeichen können hier eingegeben werden. `<input>` -Elemente werden in einem `<form>` -Element verwendet, um Eingabesteuerelemente zu deklarieren, mit denen Benutzer Daten eingeben können.

Syntax

```
<input type="text">
```

oder (ohne Angabe eines `type` mit dem Standardattribut):

```
<input>
```

Die Standardbreite einer Textfelderingabe beträgt 20 Zeichen. Dies kann durch Angabe eines Wertes für die geändert werden `size` Attribut wie folgt aus :

```
<input type="text" size="50">
```

Die `size` Attribut ist deutlich anders als eine Breite mit CSS festlegen. Die Verwendung einer Breite definiert einen bestimmten Wert (in Pixel, Prozentsatz des übergeordneten Elements usw.), dessen Eingabe immer breit sein muss. Bei Verwendung der `size` wird die zuzuordnende Breite basierend auf der verwendeten Schriftart und der Breite der Zeichen in der Regel berechnet.

Hinweis: Durch die Verwendung des `size` wird die Anzahl der Zeichen, die in das Feld eingegeben werden können, nicht von Natur aus begrenzt, sondern nur, wie breit das Feld angezeigt wird. Informationen zum Begrenzen der Länge finden Sie unter Eingabevalidierung .

Ein Eingabefeld erlaubt nur eine Textzeile. Wenn Sie für eine größere Textmenge eine mehrzeilige Texteingabe benötigen, verwenden Sie stattdessen ein `<textarea>` -Element .

Suche

5

Die Eingabetypsuche wird für die Textsuche verwendet. Bei den meisten Browsern wird ein Lupensymbol neben dem Platz für Text eingefügt

```
<input type="search" name="googlesearch">
```

DateTime (Global)

Das Eingabeelement mit einem **Typattribut** , dessen Wert " **datetime** " ist, repräsentiert ein Steuerelement zum Festlegen des Elementwerts auf einen String, der ein **globales Datum und** eine **globale Uhrzeit (mit Zeitzoneninformationen) darstellt.**

```
<fieldset>
   <p><label>Meeting time: <input type=datetime name="meeting.start"></label>
</fieldset>
```

Zulässige Attribute:

- globale Attribute
- Name
- deaktiviert
- bilden
- Art
- Autovervollständigung
- Autofokus
- Liste

- Minimal Maximal
- Schritt (Float)
- schreibgeschützt
- erforderlicher Wert

Eingabesteuerelemente online lesen:
https://riptutorial.com/de/html/topic/277/eingabesteuerelemente

Kapitel 17: Formen

Einführung

Um Eingabeelemente zu gruppieren und Daten zu senden, verwendet HTML ein Formularelement, um Eingabe- und Übergabeelemente zu kapseln. Diese Formulare behandeln das Senden der Daten in der angegebenen Methode an eine Seite, die von einem Server oder Handler behandelt wird. In diesem Thema wird die Verwendung von HTML-Formularen beim Erfassen und Senden von Eingabedaten erläutert und veranschaulicht.

Syntax

- `<form method="post|get" action="somePage.php" target="_blank|_self|_parent|_top|framename">`

Parameter

Attribut	Beschreibung
`accept-charset`	Gibt die Zeichencodierungen an, die für die Formularübermittlung verwendet werden sollen.
`action`	Gibt an, wohin die Formulardaten gesendet werden sollen, wenn ein Formular gesendet wird.
`autocomplete`	Gibt an, ob für ein Formular die automatische Vervollständigung aktiviert oder deaktiviert werden soll.
`enctype`	Gibt an, wie die Formulardaten beim Senden an den Server codiert werden sollen (nur für Methode = "post").
`method`	Gibt die HTTP-Methode an, die beim Senden von Formulardaten (POST oder GET) verwendet werden soll.
`name`	Gibt den Namen eines Formulars an.
`novalidate`	Gibt an, dass das Formular beim Senden nicht überprüft werden soll.
`target`	Gibt an, wo die Antwort angezeigt werden soll, die nach dem Senden des Formulars empfangen wird.

Bemerkungen

Das `<form>` -Element repräsentiert einen Abschnitt, der `<fieldset>` Elemente enthält (z. B. `<button>` `<fieldset>` `<input>` `<label>` `<output>` `<select>` `<textarea>`), die Informationen an einen Server

übermitteln. Es sind sowohl Start- (`<form>`) als auch Endetags (`</form>`) erforderlich.

Examples

Einreichen

Das Aktionsattribut

Das Aktionsattribut definiert die Aktion, die ausgeführt werden soll, wenn das Formular übermittelt wird. In der Regel führt dies zu einem Skript, das die übermittelten Informationen sammelt und damit arbeitet. Wenn Sie das Feld leer lassen, wird es an dieselbe Datei gesendet

```
<form action="action.php">
```

Das Methodenattribut

Das Methodenattribut wird verwendet, um die HTTP-Methode des Formulars zu definieren, die entweder GET oder POST ist.

```
<form action="action.php" method="get">
<form action="action.php" method="post">
```

Die GET-Methode wird meistens verwendet, *um* Daten abzurufen, beispielsweise um einen Beitrag anhand seiner ID oder seines Namens zu erhalten oder eine Suchabfrage zu übermitteln. Die GET-Methode hängt die Formulardaten an die im Aktionsattribut angegebene URL an.

```
www.example.com/action.php?firstname=Mickey&lastname=Mouse
```

Die POST-Methode wird verwendet, wenn Daten an ein Skript gesendet werden. Die POST-Methode hängt die Formulardaten nicht an die Aktions-URL an, sondern sendet mithilfe des Anforderungshauptteils.

Um die Daten aus dem Formular korrekt zu senden, muss ein Namensattributname angegeben werden.
Als Beispiel senden wir den Wert des Feldes und setzen seinen Namen auf *lastname* :

```
<input type="text" name="lastname" value="Mouse">
```

Weitere Attribute

```
<form action="action.php" method="post" target="_blank" accept-charset="UTF-8"
enctype="application/x-www-form-urlencoded" autocomplete="off" novalidate>

<!-- form elements -->

</form>
```

Zielattribut im Formular-Tag

Das Zielattribut gibt einen Namen oder ein Schlüsselwort an, das angibt, wo die Antwort angezeigt werden soll, die nach dem Senden des Formulars empfangen wird.

Das Zielattribut definiert einen Namen oder ein Schlüsselwort für einen Browserkontext (z. B. Tab, Fenster oder Inlineframe).

Von Tag mit einem Zielattribut:

```
<form target="_blank">
```

Attributwerte

Wert	Beschreibung
_leer	Die Antwort wird in einem neuen Fenster oder einer neuen Registerkarte angezeigt
_selbst	Die Antwort wird in demselben Frame angezeigt (dies ist die Standardeinstellung)
_Elternteil	Die Antwort wird im übergeordneten Frame angezeigt
_oben	Die Antwort wird im gesamten Hauptteil des Fensters angezeigt
Framename	Die Antwort wird in einem benannten iframe angezeigt

> Hinweis: Das Zielattribut wurde in **HTML 4.01** ***veraltet.*** Das Zielattribut wird in ***HTML5 unterstützt.***
>
> Frames und Framesets werden in ***HTML5*** nicht unterstützt. ***Daher werden die Werte _parent, _top und Framename jetzt hauptsächlich mit Iframes verwendet*** .

Dateien hochladen

Bilder und Dateien können durch das Setzen auf den Server hochgeladen / eingereicht werden `enctype` Attribut von `form` - Tag `multipart/form-data` . `enctype` gibt an, wie Formulardaten beim `enctype` an den Server verschlüsselt werden.

Beispiel

```
<form method="post" enctype="multipart/form-data" action="upload.php">
    <input type="file" name="pic" />
    <input type="submit" value="Upload" />
</form>
```

Einige Eingabefelder gruppieren

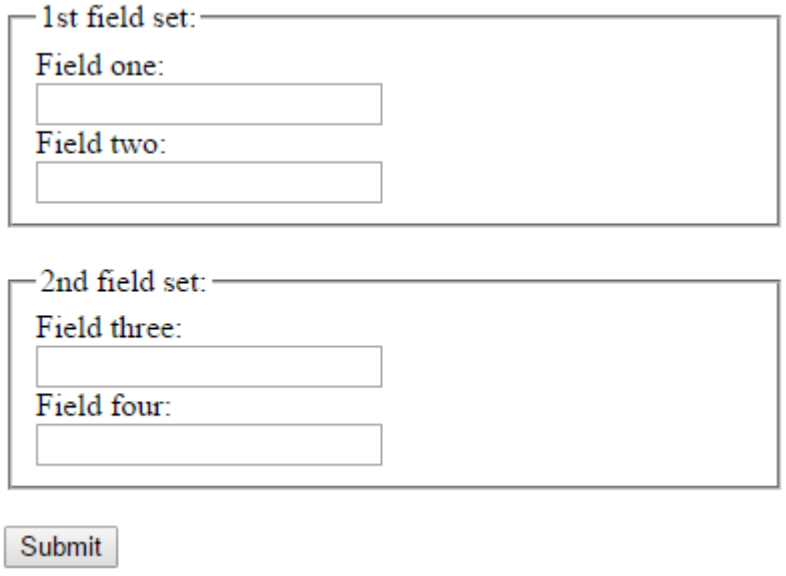
Beim Entwerfen eines Formulars möchten Sie möglicherweise einige Eingabefelder in einer Gruppe zusammenfassen, um das Formularlayout zu organisieren. Dies kann mit dem Tag erfolgen. Hier ist ein Beispiel für die Verwendung.

Sie können für jedes Feldset eine Legende für das Set festlegen, indem Sie das Tag LEGEND TEXT verwenden

Beispiel

```
<form>
  <fieldset>
     <legend>1st field set:</legend>
     Field one:<br>
     <input type="text"><br>
     Field two:<br>
     <input type="text"><br>
  </fieldset><br>
  <fieldset>
     <legend>2nd field set:</legend>
     Field three:<br>
     <input type="text"><br>
     Field four:<br>
     <input type="text"><br>
  </fieldset><br>
  <input type="submit" value="Submit">
</form>
```

Ergebnis



Browser-Unterstützung

Die neuesten Versionen von Chrome, IE, Edge, FireFox, Safari und Opera unterstützen das Tag ebenfalls

Formen online lesen: https://riptutorial.com/de/html/topic/1160/formen

Kapitel 18: Fortschrittselement

Parameter

Parameter	Wert
max	Wie viel Arbeit die Aufgabe insgesamt erfordert
Wert	Wie viel der Arbeit wurde bereits geleistet
Position	Dieses Attribut gibt die aktuelle Position des `<progress>` -Elements zurück
Etiketten	Dieses Attribut gibt eine Liste der `<progress>` -Elementbeschriftungen zurück (falls vorhanden).

Bemerkungen

Das `<progress>` -Element wird in Versionen von Internet Explorer unter 10 nicht unterstützt

Das `<progress>` -Element ist das falsche Element, das für etwas verwendet wird, das nur ein Messgerät ist, und nicht für den Taskfortschritt. Das Anzeigen der Verwendung von Speicherplatz mithilfe des Elements `<progress>` ist beispielsweise nicht geeignet. Stattdessen ist das Element `<meter>` für diese Art von Anwendungsfällen verfügbar.

Examples

Fortschritt

Das `<progress>` -Element ist neu in HTML5 und wird verwendet, um den Fortschritt einer Aufgabe darzustellen

```
<progress value="22" max="100"></progress>
```

Dadurch wird ein Balken mit 22% gefüllt.

Ändern der Farbe einer Fortschrittsleiste

Fortschrittsbalken können mit der `progress[value]` .

Dieses Beispiel gibt einem Fortschrittsbalken eine Breite von `250px` und eine Höhe von `20px`

```
progress[value] {
  width: 250px;
  height: 20px;
}
```

Fortschrittsbalken können besonders schwierig zu gestalten sein.

Chrome / Safari / Opera

Diese Browser verwenden den Selektor `-webkit-appearance` , um das Fortschritts-Tag zu `-webkit-appearance` . Um dies zu überschreiben, können wir das Erscheinungsbild zurücksetzen.

```
progress[value] {
  -webkit-appearance: none;
  appearance: none;
}
```

Nun können wir den Container selbst gestalten

```
progress[value]::-webkit-progress-bar {
  background-color: "green";
}
```

Feuerfuchs

Firefox formatiert den Fortschrittsbalken etwas anders. Wir müssen diese Stile verwenden

```
progress[value] {
  -moz-appearance: none;
  appearance: none;
  border: none;                /* Firefox also renders a border */
}
```

Internet Explorer

Internet Explorer 10+ unterstützt das `progress` . Die `background-color` Eigenschaft wird jedoch nicht unterstützt. Sie müssen stattdessen die color-Eigenschaft verwenden.

```
progress[value]  {
  -webkit-appearance: none;
     -moz-appearance: none;
          appearance: none;

  border: none;                       /* Remove border from Firefox */

  width: 250px;
  height: 20px;

  color: blue;
}
```

HTML-Fallback

Für Browser, die das `progress` nicht unterstützen, können Sie dies als Problemumgehung verwenden.

```
<progress max="100" value="20">
    <div class="progress-bar">
        <span style="width: 20%;">Progress: 20%</span>
    </div>
</progress>
```

Browser, die das Progress-Tag unterstützen, ignorieren das verschachtelte Div. Ältere Browser, die das Fortschritts-Tag nicht identifizieren können, rendern stattdessen das div.

Fortschrittselement online lesen: https://riptutorial.com/de/html/topic/5055/fortschrittselement

Kapitel 19: Fügen Sie JavaScript-Code in HTML ein

Syntax

- `<script type="text/javascript"> //some code </script>`
- `<script type="text/javascript" src="URL"></script>`
- `<script type="text/javascript" src="URL" async>//async code</script>`

Parameter

Attribut	Einzelheiten
`src`	Gibt den Pfad zu einer JavaScript-Datei an. Entweder eine relative oder absolute URL.
`type`	Gibt den MIME-Typ an. Dieses Attribut ist in HTML4 erforderlich, in HTML5 jedoch optional.
`async`	Gibt an, dass das Skript asynchron ausgeführt werden soll (nur für externe Skripts). Dieses Attribut erfordert keinen Wert (außer XHTML).
`defer`	Gibt an, dass das Skript ausgeführt werden soll, wenn die Analyse der Seite abgeschlossen ist (nur für externe Skripts). Dieses Attribut erfordert keinen Wert (außer XHTML).
`charset`	Gibt die Zeichenkodierung an, die in einer externen Skriptdatei verwendet wird, z. B. UTF-8
`crossorigin`	Wie das Element Crossorigin-Anforderungen verarbeitet
`nonce`	Kryptografisches Nonce, das in der *Inhaltssicherheitsrichtlinie verwendet wird,* überprüft CSP3

Bemerkungen

Wenn der eingebettete JavaScript-Code (Datei) verwendet wird, um http://stackoverflow.com/documentation/javascript/503/document-object-model-dom Elemente zu bearbeiten , platzieren Sie Ihre `<script></script>` -Tags direkt **vor dem schließenden `</body>` taggen** oder verwenden Sie JavaScript-Methoden oder -Bibliotheken (z. B. jQuery , um eine Vielzahl von Browsern zu verwenden), die sicherstellen, dass das DOM gelesen und für die Bearbeitung bereit ist.

Examples

Verknüpfung zu einer externen JavaScript-Datei

```
<script src="example.js"></script>
```

Das Attribut `src` funktioniert wie das Attribut href für Anker: Sie können entweder eine absolute oder eine relative URL angeben. Das obige Beispiel verweist auf eine Datei im selben Verzeichnis des HTML-Dokuments. Dies wird normalerweise in den `<head>` -Tags oben im HTML-Dokument hinzugefügt

Direkter Einschluss von JavaScript-Code

Anstatt auf eine externe Datei zu verlinken, können Sie den JS-Code auch in Ihrem HTML-Code einfügen:

```
<script>
// JavaScript code
</script>
```

Einschließen einer JavaScript-Datei, die asynchron ausgeführt wird

```
<script type="text/javascript" src="URL" async></script>
```

Umgang mit deaktiviertem Javascript

Es ist möglich, dass der Clientbrowser Javascript nicht unterstützt oder die Ausführung von Javascript deaktiviert ist, möglicherweise aus Sicherheitsgründen. Um den Benutzern mitzuteilen, dass ein Skript auf der Seite ausgeführt werden soll, kann das `<noscript>` -Tag verwendet werden. Der Inhalt von `<noscript>` wird angezeigt, wenn Javascript für die aktuelle Seite deaktiviert ist.

```
<script>
  document.write("Hello, world!");
</script>
<noscript>This browser does not support Javascript.</noscript>
```

Fügen Sie JavaScript-Code in HTML ein online lesen:
https://riptutorial.com/de/html/topic/3719/fugen-sie-javascript-code-in-html-ein

Kapitel 20: Globale Attribute

Parameter

Attribut	Beschreibung
`class`	Definiert einen oder mehrere Klassennamen für ein Element. Siehe Klassen und IDs .
`contenteditable`	Legt fest, ob der Inhalt eines Elements bearbeitet werden kann.
`contextmenu`	Definiert ein Kontextmenü, das angezeigt wird, wenn ein Benutzer mit der rechten Maustaste auf ein Element klickt.
`dir`	Legt die Textrichtung für Text innerhalb eines Elements fest.
`draggable`	Legt fest, ob ein Element gezogen werden kann.
`hidden`	Blendet ein Element aus, das derzeit nicht auf der Seite verwendet wird.
`id`	Definiert einen eindeutigen Bezeichner für ein Element. Siehe Klassen und IDs .
`lang`	Definiert die Sprache des Elementinhalts und seiner Textattributwerte. Siehe Inhaltssprachen .
`spellcheck`	Legt fest, ob der Inhalt eines Elements auf Rechtschreibung / Grammatik überprüft wird.
`style`	Definiert eine Reihe von Inline-CSS-Stilen für ein Element.
`tabindex`	Legt die Reihenfolge fest, in der Elemente auf einer Seite durch die Tabulator-Tastenkombination navigiert werden.
`title`	Definiert zusätzliche Informationen zu einem Element, im Allgemeinen in Form von QuickInfo-Text bei Mouseover.
`translate`	Legt fest, ob der Inhalt eines Elements übersetzt werden soll.

Bemerkungen

Globale Attribute werden einfach zugeordnet und können auf jedes Element im gesamten Dokument angewendet werden.

Examples

Inhalteditierbares Attribut

```
<p contenteditable>This is an editable paragraph.</p>
```

Wenn Sie auf den Absatz klicken, kann der Inhalt ähnlich einem Eingabetextfeld bearbeitet werden.

Wenn das `contenteditable` Attribut nicht für ein Element festgelegt ist, erbt das Element es von seinem übergeordneten Element. So kann auch der gesamte untergeordnete Text eines inhaltsbearbeitbaren Elements bearbeitet werden. Sie *können* ihn jedoch für bestimmten Text deaktivieren, z.

```
<p contenteditable>
  This is an editable paragraph.
  <span contenteditable="false">But not this.</span>
</p>
```

Beachten Sie, dass ein nicht editierbares Textelement in einem bearbeitbaren Element auch einen Textcursor enthält, der von seinem übergeordneten Element geerbt wird.

Globale Attribute online lesen: https://riptutorial.com/de/html/topic/2811/globale-attribute

Kapitel 21: HTML 5-Cache

Bemerkungen

Die Manifestdatei ist eine einfache Textdatei, die dem Browser mitteilt, was er zwischenspeichern soll (und was niemals). Die empfohlene Dateierweiterung für Manifestdateien lautet: ".appcache" Die Manifestdatei hat drei Abschnitte:

CACHE MANIFEST - Dateien, die unter dieser Kopfzeile aufgeführt sind, werden nach dem ersten Download zwischengespeichert

NETWORK - Dateien, die unter diesem Header aufgeführt sind, erfordern eine Verbindung zum Server und werden niemals zwischengespeichert

FALLBACK - Dateien, die unter dieser Kopfzeile aufgelistet sind, geben Fallback-Seiten an, wenn auf eine Seite nicht zugegriffen werden kann

Examples

Ein einfaches Beispiel für den HTML-5-Cache

Dies ist unsere index.html Datei

```
<!DOCTYPE html>
<html manifest="index.appcache">
<body>
    <p>Content</p>
</body>
</html>
```

Dann erstellen wir die index.appcache-Datei mit den folgenden Codes

```
CACHE MANIFEST
index.html
```

Schreiben Sie die Dateien, die zwischengespeichert werden sollen, und laden Sie index.html. Gehen Sie dann in den Offline-Modus und laden Sie die Registerkarte erneut

Hinweis: Die beiden Dateien müssen sich in diesem Beispiel im selben Ordner befinden

HTML 5-Cache online lesen: https://riptutorial.com/de/html/topic/8024/html-5-cache

Kapitel 22: HTML-Ereignisattribute

Examples

HTML-Formularereignisse

Ereignisse, die durch Aktionen in einem HTML-Formular ausgelöst werden (gilt für fast alle HTML-Elemente, wird jedoch am häufigsten in Formularelementen verwendet)

Attribut	Beschreibung
onblur	Feuert den Moment ab, an dem das Element den Fokus verliert
bei Änderung	Löst den Moment aus, wenn der Wert des Elements geändert wird
oncontextmenu	Skript, das ausgeführt werden soll, wenn ein Kontextmenü ausgelöst wird
im Fokus	Feuert den Moment ab, an dem das Element den Fokus erhält
oninput	Skript, das ausgeführt werden soll, wenn ein Element eine Benutzereingabe erhält
nicht ungültig	Skript, das ausgeführt werden soll, wenn ein Element ungültig ist
onreset	Wird ausgelöst, wenn auf die Schaltfläche "Zurücksetzen" in einem Formular geklickt wird
onsearch	Wird ausgelöst, wenn der Benutzer etwas in ein Suchfeld schreibt (für <input = "search">)
onselect	Wird ausgelöst, nachdem in einem Element Text ausgewählt wurde
onsubmit	Wird ausgelöst, wenn ein Formular gesendet wird

Tastaturereignisse

Attribut	Beschreibung
onkeydown	Wird ausgelöst, wenn ein Benutzer eine Taste drückt
onkeypress	Wird ausgelöst, wenn ein Benutzer eine Taste drückt
onkeyup	Wird ausgelöst, wenn ein Benutzer eine Taste loslässt

HTML-Ereignisattribute online lesen: https://riptutorial.com/de/html/topic/10924/html-ereignisattribute

Kapitel 23: IFrames

Parameter

Attribut	Einzelheiten
`name`	Legt den Namen des Elements fest, der zusammen mit `a` Tag verwendet werden soll, um die `src` des iframe zu ändern.
`width`	Legt die Breite des Elements in Pixel fest.
`height`	Legt die Höhe des Elements in Pixel fest.
`src`	Gibt die Seite an, die im Frame angezeigt wird.
`srcdoc`	Gibt den Inhalt an, der im Frame angezeigt wird, sofern der Browser ihn unterstützt. Der Inhalt muss gültiges HTML sein.
`sandbox`	Wenn festgelegt, wird der Inhalt des Iframes als eindeutiger Ursprung behandelt, und Funktionen wie Skripte, Plugins, Formulare und Popups werden deaktiviert. Einschränkungen können durch Hinzufügen einer durch Leerzeichen getrennten Liste von Werten selektiv gelockert werden. Mögliche Werte finden Sie in der Tabelle in Anmerkungen.
`allowfullscreen`	`requestFullscreen()` ob der Inhalt des iframe `requestFullscreen()`

Bemerkungen

Ein iframe wird verwendet, um ein anderes Dokument in das aktuelle HTML-Dokument einzubetten.

Sie können iframes für die Anzeige verwenden:

- andere HTML-Seiten in derselben Domain;
- andere HTML-Seiten in einer anderen Domain (siehe unten - Richtlinien mit gleichem Ursprung);
- PDF-Dokumente (obwohl der IE möglicherweise Probleme hat, kann diese SO-Frage hilfreich sein);

Sie sollten einen Iframe als letzten Ausweg verwenden, da er Probleme mit dem Lesezeichen und der Navigation hat und es immer bessere Optionen als einen Iframe gibt. Diese SO-Frage sollte Ihnen helfen, mehr über die Höhen und Tiefen von Iframes zu erfahren.

Gleiche Ursprungspolitik

Einige Sites können nicht mit einem iframe angezeigt werden, da sie eine Richtlinie mit der Bezeichnung " Same-Origin-Richtlinie" erzwingen. Dies bedeutet, dass sich die Site, auf der sich der iframe befindet, in derselben Domäne befinden muss wie die anzuzeigende.

Diese Richtlinie gilt auch für die Bearbeitung von Inhalten, die sich in einem iFrame befinden. Wenn der iFrame auf Inhalte von einer anderen Domäne zugreift, können Sie nicht auf den Inhalt eines iFrame zugreifen oder ihn bearbeiten.

Das iframe-Element in W3C

---

sandbox

Wenn das `sandbox` Attribut gesetzt ist, werden dem iframe zusätzliche Einschränkungen hinzugefügt. Eine durch Leerzeichen getrennte Liste von Token kann verwendet werden, um diese Einschränkungen zu lockern.

Wert	Einzelheiten
`allow-forms`	Ermöglicht die Übermittlung von Formularen.
`allow-pointer-lock`	Aktiviert die JavaScript-Zeiger-API.
`allow-popups`	Popups können mit `window.open` oder `<a target="_blank"`
`allow-same-origin`	Das iframe-Dokument verwendet seinen tatsächlichen Ursprung, anstatt einen eindeutigen zu erhalten. Bei Verwendung mit `allow-scripts` kann das iframe-Dokument das gesamte Sandboxing entfernen, wenn es aus demselben Ursprung wie das übergeordnete Dokument stammt.
`allow-scripts`	Aktiviert Skripte. Das iframe-Dokument und das übergeordnete Dokument können möglicherweise mithilfe der `postMessage()` API miteinander kommunizieren. Bei Verwendung mit " `allow-same-origin` das iframe-Dokument das gesamte Sandboxing entfernen, wenn es aus demselben Ursprung wie das übergeordnete Dokument stammt.
`allow-top-navigation`	Ermöglicht dem Inhalt des iframe, den Speicherort des Dokuments der obersten Ebene zu ändern.

Examples

Grundlagen eines Inline-Frames

Der Begriff "IFrame" steht für Inline Frame. Es kann verwendet werden, um eine weitere Seite in Ihre Seite aufzunehmen. Dies ergibt einen kleinen Rahmen, der den genauen Inhalt der `base.html` .

```
<iframe src="base.html"></iframe>
```

Rahmengröße einstellen

Die Größe des IFrame kann mithilfe der Attribute `width` und `height` geändert werden, wobei die Werte in Pixeln dargestellt werden (HTML 4.01 erlaubt Prozentwerte, HTML 5 jedoch nur Werte in CSS-Pixeln).

```
<iframe src="base.html" width="800" height="600"></iframe>
```

Verwenden von Ankern mit IFrames

Normalerweise wird eine Änderung der Webseite in einem Iframe mit dem Iframe initiiert, z. B. durch Klicken auf einen Link im Ifame. Es ist jedoch möglich, den Inhalt eines IFrame außerhalb des IFrame zu ändern. Sie können ein Anker - Tag , dessen Verwendung `href` - Attribut wird auf die gewünschte URL und deren `target` zu den wichtigsten iframe gesetzt `name` - Attribut.

```
<iframe src="webpage.html" name="myIframe"></iframe>
<a href="different_webpage.html" target="myIframe">Change the Iframe content to
different_webpage.html</a>
```

Verwenden des Attributs "srcdoc"

Das Attribut " `srcdoc` kann (anstelle des Attributs " `src`) verwendet werden, um den genauen Inhalt des iframe als gesamtes HTML-Dokument anzugeben. Dies ergibt einen IFrame mit dem Text "IFrames are cool!"

```
<iframe srcdoc="<p>IFrames are cool!</p>"></iframe>
```

Wenn das `srcdoc` Attribut nicht vom Browser unterstützt wird, `srcdoc` der `srcdoc` stattdessen auf das `src` Attribut zurück. Wenn jedoch die `src` und `srcdoc` vorhanden sind und vom Browser unterstützt werden, hat `srcdoc` Vorrang.

```
<iframe srcdoc="<p>Iframes are cool!</p>" src="base.html"></iframe>
```

Wenn der Browser das `srcdoc` Attribut nicht unterstützt, zeigt er im obigen Beispiel stattdessen den Inhalt der `base.html` Seite an.

Sandboxen

Im Folgenden wird eine nicht vertrauenswürdige Webseite mit allen aktivierten Einschränkungen eingebettet

```
<iframe sandbox src="http://example.com/"></iframe>
```

`sandbox attribute` `allow-scripts` und `allow-forms` hinzu `allow-scripts` damit die Seite Skripts ausführen und Formulare senden `sandbox attribute` .

```
<iframe sandbox="allow-scripts allow-forms" src="http://example.com/"></iframe>
```

Wenn nicht vertrauenswürdiger Inhalt (z. B. Benutzerkommentare) in derselben Domäne wie die übergeordnete Webseite vorhanden ist, kann ein Iframe verwendet werden, um Skripts zu deaktivieren, während das übergeordnete Dokument mit JavaScript dennoch mit seinem Inhalt interagieren kann.

```
<iframe sandbox="allow-same-origin allow-top-navigation"
src="http://example.com/untrusted/comments/page2">
```

Das übergeordnete Dokument kann Ereignis-Listener hinzufügen und die Größe des IFrame an den Inhalt anpassen. Zusammen mit `allow-top-navigation` kann die Sandkasten-Box als Teil des übergeordneten Dokuments erscheinen.

Diese Sandbox ist kein Ersatz für die Desinfektion von Eingaben, sondern kann als Teil einer Tiefenverteidigungsstrategie verwendet werden .

Beachten Sie auch, dass diese Sandbox von einem Angreifer unterlaufen werden kann, der einen Benutzer dazu verleitet, die Quelle des Iframes direkt zu besuchen. Der HTTP-Header für Inhaltssicherheitsrichtlinie kann verwendet werden, um diesen Angriff zu mildern.

IFrames online lesen: https://riptutorial.com/de/html/topic/499/iframes

Kapitel 24: Imagemaps

Syntax

- `<img usemap="#[map-name]">`
- `<map name="[map-name]"></map>`
- `<area>`

Parameter

Tag / Attribut	Wert
`<img>`	Nachfolgend sind die imagemap-spezifischen Attribute aufgeführt, die mit `<img>` . Es gelten reguläre `<img>` -Attribute.
`usemap`	Der `name` der Karte mit vorangestelltem Hash-Symbol. Für eine Karte mit `name="map"` sollte das Bild beispielsweise `usemap="#map"` .
`<map>`	
`name`	Der Name der Karte, um sie zu identifizieren. `usemap` mit dem `usemap` Attribut des Bildes `usemap` .
`<area>`	Nachfolgend sind `<area>` -spezifische Attribute aufgeführt. Wenn `href` angegeben wird, wobei `<area>` einem Link wird, unterstützt `<area>` auch alle Attribute des Ankertags (`<a>`) mit Ausnahme von `ping` . Sehen Sie sie in den MDN-Dokumenten .
`alt`	Der alternative Text, der angezeigt werden soll, wenn Bilder nicht unterstützt werden. Dies ist nur notwendig, wenn auch `href` im `<area>` .
`coords`	Die Koordinaten, die den auswählbaren Bereich umreißen. Wenn `shape="polygon"` , sollte dies auf eine Liste von durch Kommas getrennten "x, y" -Paaren gesetzt werden (dh `shape="polygon" coords="x1, y1, x2, y2, x3, y3, ..."`). Wenn `shape="rectangle"` , sollte dies auf `left, top, right, bottom` . Wenn `shape="circle"` , sollte dies auf `centerX, centerY, radius` .
`href`	Die URL des Hyperlinks, falls angegeben. Wenn es weggelassen wird, repräsentiert der `<area>` keinen Hyperlink.
`shape`	Die Form des `<area>` . Kann so eingestellt werden , um die `default` das gesamte Bild (keine auszuwählen `coords` Attribut erforderlich) `circle` oder `circ` für einen Kreis, `rectangle` oder `rect` für ein Rechteck und `polygon` oder `poly` für einen polygonalen Bereich , der durch Eckpunkte spezifiziert.

Bemerkungen

- Die obige Parameterliste wird aus den MDN-Dokumenten geändert: `<map>` und `<area>` .
- Es ist möglich, die Koordinaten einer Imagemap für ein Bild mit einfacheren Formen (wie im einleitenden Beispiel oben) mit einem Bildeditor zu erstellen, der Koordinaten anzeigt (wie beispielsweise GIMP). Es kann jedoch allgemein einfacher sein, einen Imagemap-Generator wie diesen zu verwenden .

Examples

Einführung in Imagemaps

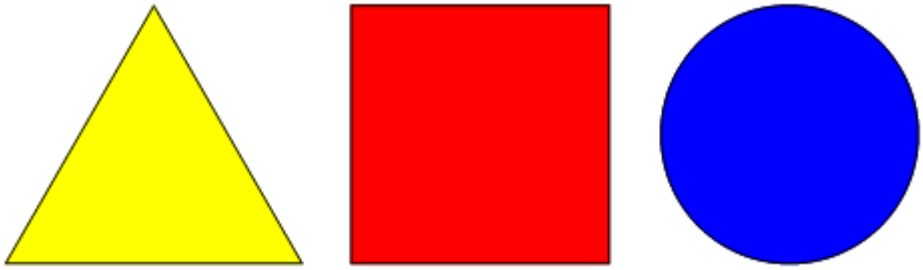
Beschreibung

Ein Imagemaps ist ein Bild mit anklickbaren Bereichen, die normalerweise als Hyperlinks dienen.

Das Bild wird durch das `<img>` -Tag definiert, und die Karte wird durch ein `<map>` -Tag mit `<area>` -Tags definiert, um jeden anklickbaren Bereich anzugeben. Verwenden Sie die Attribute `usemap` und `name` , um das Bild und die Karte zu binden.

Basisbeispiel

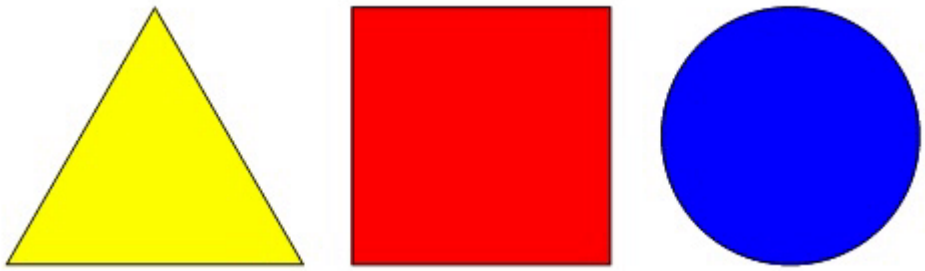
So erstellen Sie eine Imagemap, sodass jede der Formen im Bild unten anklickbar ist:



Der Code würde wie folgt lauten:

```
<img src="http://jaced.com/blogpix/2007/trisquarecircle/002.gif" usemap="#shapes">
<map name="shapes">
    <area shape="polygon" coords="79,6,5,134,153,134">
    <area shape="rectangle" coords="177,6,306,134">
    <area shape="circle" coords="397,71,65">
</map>
```

Sie sollten sehen, dass der Browser die Bereiche erkennt, wenn der Cursor zum Zeiger wird. Sehen Sie sich eine Live-Demo auf JSFiddle an, oder sehen Sie sich eine Demonstration unten an:



Imagemaps online lesen: https://riptutorial.com/de/html/topic/3819/imagemaps

Kapitel 25: Inhaltssprachen

Syntax

- `<element lang="language_code">` <! - Der Sprachcode muss im Format [ISO 639-1] (https://en.wikipedia.org/wiki/ISO_639-1) -> sein

Bemerkungen

Der Wert des `lang` Attributs muss ein gültiges **BCP 47-Sprachkennzeichen sein** oder die **leere Zeichenfolge** (wenn die Sprache unbekannt ist).

Die BCP 47- Sprach-Tags sind in der IANA-Sprach-Subtag-Registry aufgeführt .

Zugänglichkeit

Die relevanten WCAG 2.0-Erfolgskriterien sind:

- 3.1.1 Sprache der Seite
- 3.1.2 Sprache der Teile

Die verwandten WCAG 2.0-Techniken sind:

- H57: Sprachattribute für das HTML-Element verwenden
- H58: Verwendung von Sprachattributen zum Erkennen von Änderungen in der menschlichen Sprache

Examples

Elementsprache

Mit dem `lang` Attribut wird die Sprache des Elementinhalts und der Attributtextwerte angegeben:

```
<p lang="en">The content of this element is in English.</p>
```

```
<p lang="en" title="The value of this attribute is also in English.">The content of this
element is in English.</p>
```

Die Sprachdeklaration wird vererbt:

```
<div lang="en">
  <p>This element contains English content.</p>
  <p title="This attribute, too.">Same with this element.</p>
</div>
```

Elemente mit mehreren Sprachen

Sie können eine Sprachdeklaration "überschreiben":

```
<p lang="en">This English sentence contains the German word <span lang="de">Hallo</span>.</p>
```

Umgang mit Attributen mit verschiedenen Sprachen

Sie können die Sprachdeklaration eines übergeordneten Elements "überschreiben", indem Sie ein beliebiges Element außer `applet` , `base` , `basefont` , `br` , `frame` , `frameset` , `hr` , `iframe` , `meta` , `param` , `script` (von HTML 4.0) mit einem eigenen `lang` Attribut `param` :

```
<p lang="en" title="An English paragraph">
    <span lang="de" title="A German sentence">Hallo Welt!</span>
</p>
```

Basisdokumentensprache

Es empfiehlt sich, die primäre Sprache des Dokuments im `html` Element `html` :

```
<html lang="en">
```

Wenn im Dokument kein anderes `lang` Attribut angegeben ist, bedeutet dies, dass *alles* (dh Elementinhalt und Attributtextwerte) in dieser Sprache vorhanden sind.

Wenn das Dokument Teile in anderen Sprachen enthält, sollten diese Teile ihre eigenen `lang` Attribute erhalten, um die Sprachdeklaration zu "überschreiben".

Regionale URLs

Es ist möglich, das Attribut `hreflang` zu den Elementen `<a>` und `<area>` hinzuzufügen, die Hyperlinks erstellen. Dies gibt die Sprache der verknüpften Ressource an. Die definierte Sprache muss ein gültiges BCP 47 [1] -Sprachentag sein.

```
<p>
    <a href="example.org" hreflang="en">example.org</a> is one of IANA's example domains.
</p>
```

---

1. ↑ IETF-Netzwerkarbeitsgruppe: RFC 5646- *Tags zur Identifizierung von Sprachen* , IETF, September 2009

Inhaltssprachen online lesen: https://riptutorial.com/de/html/topic/737/inhaltssprachen

Kapitel 26: Klassen und IDs

Einführung

Klassen und IDs vereinfachen das Referenzieren von HTML-Elementen aus Skripts und Stylesheets. Das Klassenattribut kann für ein oder mehrere Tags verwendet werden und wird von CSS zum Gestalten verwendet. IDs sollen sich jedoch auf ein einzelnes Element beziehen, dh dieselbe ID sollte niemals zweimal verwendet werden. IDs werden im Allgemeinen mit JavaScript und internen Dokumentverknüpfungen verwendet und in CSS wird davon abgeraten. Dieses Thema enthält hilfreiche Erklärungen und Beispiele für die korrekte Verwendung von Klassen- und ID-Attributen in HTML.

Syntax

- class = "class1 class2 class3"
- id = "uniqueid"

Parameter

Parameter	Einzelheiten
Klasse	Gibt die Klasse des Elements an (nicht eindeutig)
Ich würde	Gibt die ID des Elements an (eindeutig im selben Kontext)

Bemerkungen

- Sowohl `class` als auch `id` sind globale Attribute und können daher jedem HTML-Element zugewiesen werden.
- Klassennamen müssen mit einem Buchstaben (AZ oder az) beginnen und können von Buchstaben, Ziffern, Bindestrichen und Unterstrichen gefolgt werden.
- In `HTML5` können die Klassen- und ID-Attribute für jedes Element verwendet werden. In HTML 4.0.1 waren sie gegen die Tags `<base>` , `<head>` , `<html>` , `<meta>` , `<param>` , `<script>` , `<style>` und `<title>` gesperrt.
- Ein Element kann eine oder mehrere Klassen haben. Klassen werden durch Leerzeichen getrennt und dürfen selbst keine Leerzeichen enthalten.
- Ein Element kann nur eine ID haben und muss innerhalb seines Kontexts (dh einer Webseite) eindeutig sein. IDs dürfen auch keine Leerzeichen enthalten.

Examples

Einem Element eine Klasse geben

Klassen sind Bezeichner für die Elemente, denen sie zugeordnet sind. Verwenden Sie das `class` , um einem Element eine Klasse zuzuweisen.

```
<div class="example-class"></div>
```

Um einem Element mehrere Klassen zuzuweisen, trennen Sie die Klassennamen durch Leerzeichen.

```
<div class="class1 class2"></div>
```

Klassen in CSS verwenden

Klassen können zum Gestalten bestimmter Elemente verwendet werden, ohne alle Elemente dieser Art zu ändern. Zum Beispiel können diese beiden `span` eine völlig unterschiedliche Gestaltung haben:

```
<span></span>
<span class="special"></span>
```

Klassen mit demselben Namen können einer beliebigen Anzahl von Elementen auf einer Seite zugewiesen werden, und sie erhalten alle das dieser Klasse zugeordnete Format. Dies ist immer wahr, wenn Sie das Element nicht im CSS angeben.

Zum Beispiel haben wir zwei Elemente, die beide mit der Klasse `highlight` :

```
<div class="highlight">Lorem ipsum</div>
<span class="highlight">Lorem ipsum</span>
```

Wenn unser CSS wie folgt ist, wird die Farbe Grün auf den Text in beiden Elementen angewendet:

```
.highlight { color: green; }
```

Wenn wir jedoch nur ausrichten möchten `div` ,s mit der Klasse `highlight` dann können wir Spezifität wie unten hinzufügen:

```
div.highlight { color: green; }
```

Dennoch , wenn sie mit CSS Styling, wird es in der Regel empfohlen , dass nur Klassen (zB `.highlight`) statt mit Klassen (zB Elemente verwendet werden `div.highlight`).

Wie bei jedem anderen Selektor können Klassen verschachtelt werden:

```
.main .highlight { color: red; } /* Descendant combinator */
.footer > .highlight { color: blue; } /* Child combinator */
```

Sie können den Klassenselektor auch so verketten, dass nur Elemente mit einer Kombination aus mehreren Klassen ausgewählt werden. Zum Beispiel, wenn dies unser HTML ist:

```
<div class="special left menu">This text will be pink</div>
```

Und wir möchten diesen bestimmten Text rosa färben, wir können in unserem CSS Folgendes tun:

```
.special.left.menu { color: pink; }
```

Einem Element eine ID geben

Das ID-Attribut eines Elements ist eine Kennung, die im gesamten Dokument eindeutig sein muss. Sein Zweck ist die eindeutige Identifizierung des Elements beim Verknüpfen (Verwenden eines Ankers), Skripting oder Styling (mit CSS).

```
<div id="example-id"></div>
```

Sie sollten nicht zwei Elemente mit derselben ID in demselben Dokument enthalten, auch wenn die Attribute zwei verschiedenen Arten von Elementen zugeordnet sind. Der folgende Code ist beispielsweise falsch:

```
<div id="example-id"></div>
<span id="example-id"></span>
```

Browser geben ihr Bestes, um diesen Code zu rendern. Es kann jedoch zu unerwartetem Verhalten kommen, wenn mit CSS gestylt oder mit JavaScript Funktionalität hinzugefügt wird.

Um Elemente anhand ihrer ID in CSS zu referenzieren, müssen Sie der ID ein `#` voranstellen.

```
#example-id { color: green; }
```

Um zu einem Element mit einer ID auf einer bestimmten Seite zu springen, hängen Sie `#` mit dem Elementnamen in der URL an.

```
http://example.com/about#example-id
```

Diese Funktion wird in den meisten Browsern unterstützt und erfordert kein zusätzliches JavaScript oder CSS.

Probleme im Zusammenhang mit doppelten IDs

Wenn Sie mehr als ein Element mit derselben ID haben, ist das Problem schwer zu beheben. Der HTML-Parser versucht normalerweise, die Seite auf jeden Fall zu rendern. Normalerweise tritt kein Fehler auf. Das Tempo könnte jedoch zu einer fehlerhaften Webseite führen.

In diesem Beispiel:

```
<div id="aDiv">a</div>
<div id="aDiv">b</div>
```

CSS-Selektoren funktionieren immer noch

```
#aDiv {
    color: red;
}
```

JavaScript kann jedoch nicht beide Elemente verarbeiten:

```
var html = document.getElementById("aDiv").innerHTML;
```

In diesem Fall trägt die `html` Variable nur den ersten `div` Inhalt `("a")` .

Akzeptable Werte

Für eine ID

5

Die einzigen Einschränkungen für den Wert einer `id` sind:

1. Es muss im Dokument eindeutig sein
2. Es darf keine Leerzeichen enthalten
3. Es muss mindestens ein Zeichen enthalten

Der Wert kann also aus allen Ziffern bestehen, nur einer Ziffer, nur Satzzeichen, Sonderzeichen oder was auch immer. Nur kein Leerzeichen.

Also diese sind gültig:

```
<div id="container"> ... </div>
<div id="999"> ... </div>
<div id="#%LV-||"> ... </div>
<div id="____V"> ... </div>
<div id="□□"> ... </div>
<div id="♥"> ... </div>
<div id="{}"> ... </div>
<div id="©"> ... </div>
<div id="⌂₩¤☆€~¥"> ... </div>
```

Das ist ungültig:

```
<div id=" "> ... </div>
```

Dies ist auch ungültig, wenn es in demselben Dokument enthalten ist:

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

4.01

Ein `id` Wert muss mit einem Buchstaben beginnen, dem dann nur folgen kann:

- Buchstaben (AZ / az)
- Ziffern (0-9)
- Bindestriche ("-")
- Unterstriche ("_")
- Doppelpunkte (":")
- Zeiträume (".")

Bezüglich der ersten Gruppe von Beispielen im obigen HTML5-Abschnitt ist nur eines gültig:

```
<div id="container"> ... </div>
```

Diese sind auch gültig:

```
<div id="sampletext"> ... </div>
<div id="sample-text"> ... </div>
<div id="sample_text"> ... </div>
<div id="sample:text"> ... </div>
<div id="sample.text"> ... </div>
```

Wenn es nicht mit einem Buchstaben (Groß- oder Kleinbuchstaben) beginnt, ist es nicht gültig.

Für eine Klasse

Die Regeln für Klassen sind im Wesentlichen die gleichen wie für eine `id` . Der Unterschied besteht darin , dass `class` brauchen *nicht* im Dokument eindeutig zu sein.

Verweisen Sie auf die obigen Beispiele, obwohl dies in demselben Dokument nicht gültig ist:

```
<div id="results"> ... </div>
<div id="results"> ... </div>
```

Das ist vollkommen in Ordnung:

```
<div class="results"> ... </div>
<div class="results"> ... </div>
```

Wichtiger Hinweis: Wie ID- und Klassenwerte außerhalb von HTML behandelt werden

Beachten Sie, dass die obigen Regeln und Beispiele im Kontext von HTML gelten.

Die Verwendung von Zahlen, Satzzeichen oder Sonderzeichen im Wert einer `id` oder `class` kann in anderen Kontexten wie CSS, JavaScript und regulären Ausdrücken zu Problemen führen.

Beispielsweise ist die folgende `id` in HTML5 gültig:

```
<div id="9lions"> ... </div>
```

... es ist in CSS ungültig:

4.1.3 Zeichen und Fall

> In CSS können *Bezeichner* (einschließlich Elementnamen, Klassen und IDs in Selektoren) nur die Zeichen [a-zA-Z0-9] und ISO 10646-Zeichen U + 00A0 und höher sowie den Bindestrich (-) und den Unterstrich (_); ***Sie können nicht mit einer Ziffer, zwei Bindestrichen oder einem Bindestrich gefolgt von einer Ziffer beginnen*** .
> (Betonung hinzugefügt)

In den meisten Fällen können Sie Zeichen in Kontexten, in denen sie Einschränkungen oder eine spezielle Bedeutung haben, maskieren.

W3C-Referenzen

- 3.2.5.1 Das `id` Attribut
- 3.2.5.7 Das `class`
- 6.2 SGML-Basistypen

Klassen und IDs online lesen: https://riptutorial.com/de/html/topic/586/klassen-und-ids

Kapitel 27: Leere Elemente

Einführung

Nicht alle HTML-Tags haben dieselbe Struktur. Während für die meisten Elemente ein öffnendes Tag, ein schließendes Tag und Inhalt erforderlich sind, benötigen einige Elemente - so genannte Void-Elemente - nur ein öffnendes Tag, da sie selbst keine Elemente enthalten. In diesem Thema wird die ordnungsgemäße Verwendung von leeren Elementen in HTML erläutert und veranschaulicht

Bemerkungen

Ein ungültiges Element kann keinen Inhalt haben, kann aber Attribute haben. Leere Elemente schließen sich selbst, sodass sie kein schließendes Tag haben dürfen.

In HTML5 sind die folgenden Elemente ungültig:

- `area`
- `base`
- `br`
- `col`
- `embed`
- `hr`
- `img`
- `input`
- `keygen`
- `link`
- `meta`
- `param`
- `source`
- `track`
- `wbr`

Examples

Leere Elemente

HTML 4.01 / XHTML 1.0 Strict enthält die folgenden ungültigen Elemente:

- `area` - anklickbarer, definierter Bereich in einem Bild
- `base` - gibt eine Basis-URL an, von der alle Links ausgehen
- `br` - line break
- `col` - Spalte in einer Tabelle [veraltet]
- `hr` - horizontale Linie (Linie)
- `img` - Bild
- `input` , in das Benutzer Daten eingeben
- `link` - Verknüpft eine externe Ressource mit dem Dokument

- `meta` - enthält Informationen zum Dokument
- `param` - `param` Parameter für Plugins

HTML 5-Standards enthalten alle nicht veralteten Tags aus der vorherigen Liste und

- `command` - stellt einen Befehl dar, den Benutzer aufrufen können [veraltet]
- `keygen` - vereinfacht die Generierung öffentlicher Schlüssel für Webzertifikate [veraltet]
- `source` - Gibt Medienquellen für `picture` , `audio` und `video`

Das folgende Beispiel enthält **keine** leeren Elemente:

```
<div>
    <a href="http://stackoverflow.com/">
        <h3>Click here to visit <i>Stack Overflow!</i></h3>
    </a>
    <button onclick="alert('Hello!');">Say Hello!</button>
    <p>My favorite language is <b>HTML</b>. Here are my others:</p>
    <ol>
        <li>CSS</li>
        <li>JavaScript</li>
        <li>PHP</li>
    </ol>
</div>
```

Beachten Sie, dass jedes Element ein öffnendes Tag, ein schließendes Tag und Text oder andere Elemente in den öffnenden und schließenden Tags hat. Void-Tags werden jedoch im folgenden Beispiel gezeigt:

```
<img src="https://cdn.sstatic.net/Sites/stackoverflow/company/img/logos/so/so-icon.png" />
<br>
<hr>
<input type="number" placeholder="Enter your favorite number">
```

Mit Ausnahme des img-Tags haben alle diese leeren Elemente nur ein öffnendes Tag. Das img-Tag hat im Gegensatz zu allen anderen Tags ein selbstschließendes `/` vor dem Größer-als-Zeichen des öffnenden Tags. Es wird empfohlen, vor dem Schrägstrich ein Leerzeichen zu haben.

Leere Elemente online lesen: https://riptutorial.com/de/html/topic/1449/leere-elemente

Kapitel 28: Listen

Einführung

HTML bietet drei Möglichkeiten zum Angeben von Listen: geordnete Listen, ungeordnete Listen und Beschreibungslisten. Sortierte Listen verwenden Ordnungsfolgen, um die Reihenfolge der Listenelemente anzugeben, ungeordnete Listen verwenden ein definiertes Symbol, z. B. ein Aufzählungszeichen, um Elemente in keiner festgelegten Reihenfolge aufzulisten, und Beschreibungslisten verwenden Einzüge, um Elemente mit ihren untergeordneten Elementen aufzulisten. In diesem Thema wird die Implementierung und Kombination dieser Listen in HTML-Markup erläutert.

Syntax

- `<ol> ordered list items </ol>`
- `<ul> unordered list items </ul>`
- `<li> list item (ordered and not) </li>`
- `<dl> description list items </dl>`
- `<dt> description list title </dt>`
- `<dd> description list description </dd>`

Bemerkungen

Siehe auch

Sie können eine Liste-style-type CSS - Eigenschaft zu einem hinzufügen `<ul>` -Tag zu ändern , um welche Art von Symbol verwendet wird , wird jedes Listenelement zu markieren, zum Beispiel `<ul style="list-style-type:disc">` . Die folgenden Listenstil-Typen sind zulässig:

- "list-style-type: disc" ist der Standardpunkt.
- "Listenart-Typ: Kreis" ist ein nicht ausgefüllter Kreis.
- "list-style-type: square" ist ein gefülltes Quadrat.
- "list-style-type: none" verwendet keine Markierung.

Sie können einem `<ol>` -Tag auch ein Typattribut hinzufügen, um die Art der Nummerierung zu ändern, z. B. `<ol type="1">` . Folgende Typen sind erlaubt:

- Typ = "1" ist das Standardformular.
- type = "A" verwendet Großbuchstaben in alphabetischer Reihenfolge
- type = "a" verwendet Kleinbuchstaben in alphabetischer Reihenfolge
- type = "I" verwendet römische Ziffern mit Großbuchstaben
- type = "i" verwendet römische Ziffern mit Kleinbuchstaben

Examples

Ungeordnete Liste

Eine ungeordnete Liste kann mit dem Tag `<ul>` und jedes Listenelement kann mit dem Tag `<li>` wie im folgenden Beispiel gezeigt:

```
<ul>
  <li>Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ul>
```

Dadurch wird eine Liste mit Aufzählungszeichen (Standardstil) erstellt:

- Artikel
- Ein anderer Artikel
- Noch ein weiterer Artikel

> Sie sollten `ul` , um eine Liste von Elementen anzuzeigen, bei denen die Reihenfolge der Elemente nicht wichtig ist. Wenn die Liste durch Ändern der Reihenfolge falsch ist, sollten Sie `<ol>` .

Bestellliste

Mit dem `<ol>` -Tag kann eine geordnete Liste erstellt werden. Jedes Listenelement kann mit dem `<li>` -Tag wie im folgenden Beispiel erstellt werden:

```
<ol>
  <li>Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

Dadurch wird eine nummerierte Liste erstellt (Standardeinstellung):

1. Artikel
2. Ein anderer Artikel
3. Noch ein weiterer Artikel

Manuelles Ändern der Zahlen

Es gibt verschiedene Möglichkeiten, wie Sie mit den Nummern auf den Listenelementen einer geordneten Liste spielen können. Die erste Möglichkeit besteht darin, eine Startnummer mithilfe des `start` festzulegen. Die Liste beginnt bei dieser definierten Nummer und wird wie gewohnt um eins erhöht.

```
<ol start="3">
  <li>Item</li>
  <li>Some Other Item</li>
  <li>Yet Another Item</li>
```

```
</ol>
```

Dadurch wird eine nummerierte Liste erstellt (Standardeinstellung):

3. Artikel
4. Einige andere Artikel
5. Noch ein weiterer Artikel

Sie können einen bestimmten Listeneintrag auch explizit auf eine bestimmte Nummer setzen. Weitere Listenelemente nach einem mit einem bestimmten Wert werden um einen weiteren Wert von diesem Listenelement erhöht, wobei die Position der übergeordneten Liste ignoriert wird.

```
<li value="7"></li>
```

Es ist auch erwähnenswert, dass Sie durch Verwendung des `value` Attributs direkt für ein Listenelement das vorhandene Nummerierungssystem einer geordneten Liste überschreiben können, indem Sie die Nummerierung mit einem niedrigeren Wert erneut starten. Wenn also die übergeordnete Liste bereits den Wert 7 erreicht hat und ein Listenelement mit dem Wert 4 gefunden wurde, wird dieses Listenelement immer noch als 4 angezeigt und von diesem Punkt an weiter gezählt.

```
<ol start="5">
  <li>Item</li>
  <li>Some Other Item</li>
  <li value="4">A Reset Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

Im obigen Beispiel wird also eine Liste erstellt, die dem Nummerierungsmuster von 5, 6, 4, 5, 6 folgt - beginnend mit einer niedrigeren Nummer als der vorherigen und mit der Nummer 6 in der Liste.

Hinweis: Die `start` und `value` - Attribute akzeptieren nur eine Zahl - auch wenn die geordnete Liste gesetzt wird , wie römische Ziffern oder Buchstaben angezeigt werden soll .

5

Sie können die Nummerierung `reversed` indem Sie Ihrem `ol` Element einen umgekehrten Code hinzufügen:

```
<ol reversed>
  <li>Item</li>
  <li>Some Other Item</li>
  <li value="4">A Reset Item</li>
  <li>Another Item</li>
  <li>Yet Another Item</li>
</ol>
```

Die umgekehrte Nummerierung ist hilfreich, wenn Sie kontinuierlich zu einer Liste hinzufügen, z.

B. mit neuen Podcast-Episoden oder Präsentationen, und Sie möchten, dass die neuesten Elemente zuerst angezeigt werden.

Die Art der Ziffer ändern

Sie können den Typ der in der Listenelementmarkierung angezeigten Zahl einfach mit dem `type` Attribut ändern

```
<ol type="1|a|A|i|I">
```

Art	Beschreibung	Beispiele
`1`	Standardwert - Dezimalzahlen	1,2,3,4
`a`	Alphabetisch geordnet (Kleinbuchstaben)	A B C D
`A`	Alphabetisch geordnet (Großbuchstaben)	A B C D
`i`	Römische Ziffern (Kleinbuchstaben)	i, ii, iii, iv
`I`	Römische Ziffern (Großbuchstaben)	I, II, III, IV

> Sie sollten `ol` , um eine Liste von Artikeln anzuzeigen, bei denen die Artikel absichtlich bestellt wurden und die Reihenfolge hervorgehoben werden sollte. Wenn das Ändern der Reihenfolge der Elemente die Liste NICHT falsch macht, sollten Sie `<ul>` .

Beschreibungsliste

Eine Beschreibungsliste (oder *Definitionsliste* , wie sie vor HTML5 aufgerufen wurde) kann mit dem `dl` Element erstellt werden. Es besteht aus Name-Wert-Gruppen, wobei der Name im `dt` Element und der Wert im `dd` Element angegeben ist.

```
<dl>
  <dt>name 1</dt>
  <dd>value for 1</dd>
  <dt>name 2</dt>
  <dd>value for 2</dd>
</dl>
```

Live-Demo

Eine Name-Wert-Gruppe kann mehr als einen Namen und / oder mehr als einen Wert haben (die Alternativen darstellen):

```
<dl>

  <dt>name 1</dt>
  <dt>name 2</dt>
  <dd>value for 1 and 2</dd>
```

```
  <dt>name 3</dt>
  <dd>value for 3</dd>
  <dd>value for 3</dd>

</dl>
```

Live-Demo

Verschachtelte Listen

Sie können Listen verschachteln, um Unterelemente eines Listenelements darzustellen.

```
<ul>
  <li>item 1</li>
  <li>item 2
    <ul>
      <li>sub-item 2.1</li>
      <li>sub-item 2.2</li>
    </ul>
  </li>
  <li>item 3</li>
</ul>
```

- Gegenstand 1
- Punkt 2
  - Unterposition 2.1
  - Unterposition 2.2
- Punkt 3

Die verschachtelte Liste muss ein Kind des Elements `li` .

Sie können auch verschiedene Arten von Listen verschachteln:

```
<ol>
    <li>Hello, list!</li>
    <li>
        <ul>
            <li>Hello, nested list!</li>
        </ul>
    </li>
</ol>
```

Listen online lesen: https://riptutorial.com/de/html/topic/393/listen

Kapitel 29: Markierungszitate

Bemerkungen

`cite` und `blockquote` Elemente sollten für den Zweck nicht ein Gespräch zu vertreten, Transkripte von Gesprächen, Dialoge in Skripten, Aufzeichnungen von Instant Messages und anderen Situationen , in denen verschiedenen Spielern abwechselnd in der Sprache verwendet werden.

Examples

Inline mit

Das **Element** `q` kann für ein Zitat verwendet werden, das Teil eines Satzes ist:

```
<p>She wrote <q>The answer is 42.</q> and everyone agreed.</p>
```

Anführungszeichen

4.01

Anführungszeichen sollten nicht hinzugefügt werden. Benutzeragenten sollten (in HTML 4.01) resp. muss (in HTML 4.0) automatisch gerendert werden.

5

Anführungszeichen dürfen nicht hinzugefügt werden. Benutzeragenten rendern sie automatisch.

Quell-URL (`cite`)

Das **Attribut** `cite` kann verwendet werden, um auf die URL der zitierten Quelle zu verweisen:

```
<p>She wrote <q cite="http://example.com/blog/hello-world">The answer is 42.</q> and everyone
agreed.</p>
```

Beachten Sie, dass Browser diese URL normalerweise nicht anzeigen. Wenn die Quelle relevant ist, sollten Sie zusätzlich einen Hyperlink (`a` Element) hinzufügen.

Blockieren mit

Das `blockquote` **Element** kann für ein (Block-Level) -Zitat verwendet werden:

```
<blockquote>
  <p>The answer is 42.</p>
</blockquote>
```

Quell-URL (`cite`)

Das **Attribut** `cite` kann verwendet werden, um auf die URL der zitierten Quelle zu verweisen:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
</blockquote>
```

Beachten Sie, dass Browser diese URL normalerweise nicht anzeigen. Wenn die Quelle relevant ist, sollten Sie zusätzlich einen Hyperlink (`a` Element) hinzufügen (siehe Abschnitt *Zitation / Attribution,* wo dieser Link platziert werden soll).

Zitierung / Namensnennung

4.01

Die Zitierung / Attribution sollte nicht Teil des `blockquote` Elements sein:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
</blockquote>
<p>Source: <cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
```

Sie können ein `div` Element hinzufügen, um das Zitat und das Zitat zu gruppieren, es besteht jedoch keine Möglichkeit, sie semantisch zu verknüpfen.

Das **Element** `cite` kann für die Referenz der zitierten Quelle verwendet werden (nicht jedoch für den Namen des Autors).

5

Das Zitat / Zuschreibung (zB der Hyperlink die Quell - URL zu geben) kann innerhalb der seine `blockquote` , aber in diesem Fall muss er innerhalb eines seines `cite` Elements (für in-Text Zuschreibungen) oder `footer` Element:

```
<blockquote cite="http://example.com/blog/hello-world">
  <p>The answer is 42.</p>
  <footer>
    <p>Source: <cite><a href="http://example.com/blog/hello-world" rel="external">Hello
World</a></cite></p>
  </footer>
</blockquote>
```

Das **Element** `cite` kann für die Referenz der zitierten Quelle oder für den Namen des Autors des Zitats verwendet werden.

Markierungszitate online lesen: https://riptutorial.com/de/html/topic/2943/markierungszitate

Kapitel 30: Medienelemente

Parameter

Attribut	Einzelheiten
Breite	Legt die Breite des Elements in Pixel fest.
Höhe	Legt die Höhe des Elements in Pixel fest.
`<source>`	Definiert Ressourcen für Audio- oder Videodateien
Spur	Definiert die Textspur für Medienelemente
Kontrollen	Zeigt Steuerelemente an
automatisches Abspielen	Starten Sie die Wiedergabe der Medien automatisch
Schleife	Spielt die Medien in einem wiederholten Zyklus ab
stummgeschaltet	Spielt die Medien ohne Ton ab
Poster	Weist ein Bild zur Anzeige zu, bis ein Video geladen ist

Bemerkungen

Unterstützung in Browsern

Merkmal	Chrom	Firefox (Gecko)	Internet Explorer	Oper	Safari
Grundlegende Unterstützung	3,0	3,5 (1.9.1)	9,0	10,50	3.1
`<audio>` : PCM in WAVE	(Ja)	3,5 (1.9.1)	Keine Unterstützung	10,50	3.1
`<audio>` : Vorbis in WebM	(Ja)	4,0 (2,0)	Keine Unterstützung	10,60	3.1
`<audio>` : Streaming von Vorbis / Opus in	?	36,0	?	?	?

Merkmal	Chrom	Firefox (Gecko)	Internet Explorer	Oper	Safari
WebM über MSE					
`<audio>` : Vorbis in Ogg	(Ja)	3,5 (1.9.1)	Keine Unterstützung	10,50	Keine Unterstützung
`<audio>` : MP3	(Ja)	(Ja)	9,0	(Ja)	3.1
`<audio>` : MP3 in MP4	?	?	?	?	(Ja)
`<audio>` : AAC in MP4	(Ja)	(Ja)	9,0	(Ja)	3.1
`<audio>` : Opus in Ogg	27,0	15,0 (15,0)	?	?	?
`<video>` : VP8 und Vorbis in WebM	6,0	4,0 (2,0)	9,0	10,60	3.1
`<video>` : VP9 und Opus in WebM	29,0	28,0 (28,0)	?	(Ja)	?
`<video>` : Streaming von WebM über MSE	?	42,0 (42,0)	?	?	?
`<video>` : Theora und Vorbis in Ogg	(Ja)	3,5 (1.9.1)	Keine Unterstützung	10,50	Keine Unterstützung
`<video>` : H.264 und MP3 in MP4	(Ja)	(Ja)	9,0	(Ja)	(Ja)
`<video>` : H.264 und AAC in MP4	(Ja)	(Ja)	9,0	(Ja)	3.1
irgendein anderes Format	Keine Unterstützung	Keine Unterstützung	Keine Unterstützung	Keine Unterstützung	3.1

Examples

Verwenden von ` `und` `Element zum Anzeigen von Audio- / Videoinhalten

Verwenden Sie das HTML- oder `<audio>` -Element, um Video- / Audio-Inhalte in ein Dokument einzubetten. Das Video- / Audioelement enthält eine oder mehrere Video- / Audioquellen. Verwenden Sie zum Angeben einer Quelle entweder das Attribut src oder das Element `<source>` . Der Browser wählt den am besten geeigneten aus.

Audio-Tag-Beispiel:

```
<!-- Simple video example -->
<video src="videofile.webm" autoplay poster="posterimage.jpg">
  Sorry, your browser doesn't support embedded videos,
  but don't worry, you can <a href="videofile.webm">download it</a>
  and watch it with your favorite video player!
</video>

<!-- Video with subtitles -->
<video src="foo.webm">
  <track kind="subtitles" src="foo.en.vtt" srclang="en" label="English">
  <track kind="subtitles" src="foo.sv.vtt" srclang="sv" label="Svenska">
</video>
<!-- Simple video example -->
<video width="480" controls poster="https://archive.org/download/WebmVp8Vorbis/webmvp8.gif" >
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.webm" type="video/webm">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8_512kb.mp4" type="video/mp4">
  <source src="https://archive.org/download/WebmVp8Vorbis/webmvp8.ogv" type="video/ogg">
  Your browser doesn't support HTML5 video tag.
</video>
```

Audio-Tag-Beispiel:

```
<!-- Simple audio playback -->
<audio src="http://developer.mozilla.org/@api/deki/files/2926/=AudioTest_(1).ogg" autoplay>
  Your browser does not support the <code>audio</code> element.
</audio>

<!-- Audio playback with captions -->
<audio src="foo.ogg">
  <track kind="captions" src="foo.en.vtt" srclang="en" label="English">
  <track kind="captions" src="foo.sv.vtt" srclang="sv" label="Svenska">
</audio>
```

Audio

HTML5 bietet einen neuen Standard zum Einbetten einer Audiodatei in eine Webseite.

Sie können eine Audiodatei mit dem `<audio>` -Element in eine Seite einbetten:

```
<audio controls>
  <source src="file.mp3" type="audio/mpeg">
Your browser does not support the audio element.
```

```
</audio>
```

Video

Sie können auch ein Video mit dem Element `<video>` in eine Webseite einbetten:

```
<video width="500" height="700" controls>
  <source src="video.mp4" type="video/mp4">
Your browser does not support the video tag.
</video>
```

Video-Header oder Hintergrund

Hinzufügen eines Videos, das automatisch in einer Schleife abgespielt wird und keine Steuerelemente oder Sounds enthält. Perfekt für einen Video-Header oder Hintergrund.

```
<video width="1280" height="720" autoplay muted loop poster="video.jpg" id="videobg">
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  <source src="video.ogg" type="video/ogg">
</video>
```

Dieses CSS bietet einen Fallback, wenn das Video nicht geladen werden kann. Beachten Sie, dass es empfohlen wird, das erste Bild des Videos als Poster video.jpg zu verwenden.

```
#videobg {
  background: url(video.jpg) no-repeat;
  background-size: cover;
}
```

Medienelemente online lesen: https://riptutorial.com/de/html/topic/2111/medienelemente

Kapitel 31: Meta-Informationen

Einführung

Metatags in HTML-Dokumenten enthalten nützliche Informationen über das Dokument, einschließlich Beschreibung, Schlüsselwörter, Autor, Änderungsdatum und rund 90 weitere Felder. Dieses Thema behandelt die Verwendung und den Zweck dieser Tags.

Syntax

- `<meta name="metadata name" content="value">`
- `<meta http-equiv="pragma directive" content="value">`
- `<meta charset="encoding label">`

Bemerkungen

Das Meta-Tag ist ein HTML-Tag, mit dem die Metadaten des HTML-Dokuments festgelegt werden. Meta-Tags müssen in das head-Element eingefügt werden. Eine Seite kann eine beliebige Anzahl von Metatags haben.

Die Meta-Tag- `keywords` normalerweise nicht von Robotern verwendet. Die meisten Suchmaschinen bestimmen, welche Schlüsselwörter zum Inhalt der Webseiten passen. Nichts besagt, dass Sie das Meta-Tag "Keywords" nicht mehr enthalten sollten.

Die Metadaten einer Seite werden hauptsächlich vom Browser verwendet (z. B. die Skalierung eines Dokuments) und von Web-Crawler-Spidern von Suchmaschinen (Google, Yahoo! Bing).

Die Spezifikation enthält eine Reihe standardisierter Metadatennamen zur Verwendung mit `<meta name>` und standardisierte Metadaten-Pragma-Direktiven zur Verwendung mit `<meta http-equiv>` . Viele Dienste im Internet (Webcrawler, Authoring-Tools, Social-Sharing-Dienste usw.) verwenden jedoch das `<meta name>` -Formular als generischen Erweiterungspunkt für Metadaten. Einige davon sind auf der Wiki-Seite der Spec aufgelistet.

Examples

Zeichenkodierung

Das `charset` gibt die Zeichenkodierung für das HTML-Dokument an und muss eine gültige Zeichenkodierung sein (Beispiele umfassen `windows-1252` , `ISO-8859-2` , `Shift_JIS` und `UTF-8`). `UTF-8` (Unicode) wird am häufigsten verwendet und sollte für jedes neue Projekt verwendet werden.

5

```
<meta charset="UTF-8">
```

```
<meta charset="ISO-8859-1">
```

Alle Browser haben das `<meta charset>` immer erkannt. Wenn Sie jedoch aus irgendeinem Grund Ihre Seite als gültiges HTML 4.01 benötigen, können Sie stattdessen Folgendes verwenden:

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

```
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
```

Siehe auch Encoding Standard , um alle verfügbaren Zeichenkodierungsetiketten anzuzeigen, die von Browsern erkannt werden.

Automatisches Aktualisieren

Um die Seite alle fünf Sekunden zu aktualisieren, fügen Sie dieses `meta` im `head` Element hinzu:

```
<meta http-equiv="refresh" content="5">
```

VORSICHT! Dies ist zwar ein gültiger Befehl, es wird jedoch empfohlen, dass Sie ihn nicht verwenden, da dies negative Auswirkungen auf die Benutzererfahrung hat. Ein zu häufiges Aktualisieren der Seite kann dazu führen, dass sie nicht mehr reagiert und häufig zum Anfang der Seite verschoben wird. Wenn einige Informationen auf der Seite ständig aktualisiert werden müssen, gibt es wesentlich bessere Möglichkeiten, dies zu tun, indem nur ein Teil einer Seite aktualisiert wird.

Mobile Layoutsteuerung

Häufig für Mobilgeräte optimierte Websites verwenden das Tag `<meta name="viewport">` wie folgt:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Das `viewport` Element gibt dem Browser Anweisungen zum Steuern der Dimensionen und der Skalierung der Seite auf der Grundlage des von Ihnen verwendeten Geräts.

In dem obigen Beispiel `content="width=device-width` bedeutet , dass der Browser die Breite der Seite in der Breite seines eigenen Bildschirm machen. Also , wenn das Bildschirm ist `480px wide` , wird das Browser - Fenster seines `480px wide` . `initial-scale=1` zeigt, dass der anfängliche Zoom (der in diesem Fall 1 ist, bedeutet, dass er nicht zoomt).

Folgende Attribute werden von diesem Tag unterstützt:

Attribut	Beschreibung
`width`	Die Breite des virtuellen Ansichtsfensters des Geräts. Werte [1] : `device-width` oder tatsächliche Breite in Pixel, z. B. `480`
`height`	Die Höhe des virtuellen Ansichtsfensters des Geräts.

Attribut	Beschreibung
	Werte [2] : `device-height` oder tatsächliche Breite in Pixel, z. B. `600`
`initial-scale`	Der anfängliche Zoom beim Laden der Seite. 1.0 zoomt nicht.
`minimum-scale`	Der Mindestbetrag, den der Besucher auf die Seite zoomen kann. `1.0` zoomt nicht.
`maximum-scale`	Der maximale Betrag, den der Besucher auf die Seite zoomen kann. `1.0` zoomt nicht.
`user-scalable`	Ermöglicht das Vergrößern und Verkleinern des Geräts. Werte sind `yes` oder `no` . Wenn Nein eingestellt ist, kann der Benutzer die Webseite nicht zoomen. Der Standardwert ist yes. Browsereinstellungen können diese Regel ignorieren.

Anmerkungen:

[1] Die `width` -Eigenschaft kann entweder in *Pixel* (`width=600`) oder nach *Gerätebreite* (`width=device-width`) angegeben werden, die die physische Breite des Bildschirms des Geräts darstellt.

[2] Ebenso kann die `height` -Eigenschaft entweder in `pixels` (`height=600`) oder nach `device-height` (`height=device-height`) angegeben werden, die die physische Höhe des Bildschirms des Geräts darstellt.

Informationen zur Seite

`application-name`

Geben Sie den Namen der Webanwendung an, die die Seite darstellt.

```
<meta name="application-name" content="OpenStreetMap">
```

Wenn es sich nicht um eine Webanwendung handelt, darf das Meta-Tag `application-name` nicht verwendet werden.

`author`

Stellen Sie den Autor der Seite ein:

```
<meta name="author" content="Your Name">
```

Es kann nur ein Name vergeben werden.

`description`

Stellen Sie die Beschreibung der Seite ein:

```
<meta name="description" content="Page Description">
```

Das `description` Meta-Tag kann von verschiedenen Suchmaschinen verwendet werden, während Sie Ihre Webseite für Suchzwecke indizieren. Normalerweise ist die Beschreibung des Meta-Tags die kurze Zusammenfassung, die unter dem Haupttitel der Seite / Website in den Suchmaschinenergebnissen angezeigt wird. Google verwendet normalerweise nur die ersten 20-25 Wörter Ihrer Beschreibung.

generator

```
<meta name="generator" content="HTML Generator 1.42">
```

Gibt eines der Software-Pakete an, die zum Generieren des Dokuments verwendet werden. Nur für Seiten, auf denen das Markup automatisch generiert wird.

keywords

Festlegen von Schlüsselwörtern für Suchmaschinen (durch Kommas getrennt):

```
<meta name="keywords" content="Keyword1, Keyword2">
```

Das `keywords` Meta-Tag wird manchmal von Suchmaschinen verwendet, um die für Ihre Webseite relevante Suchanfrage zu kennen.
Als Daumenregel ist es wahrscheinlich eine gute Idee, nicht zu viele Wörter hinzuzufügen, da die meisten Suchmaschinen, die dieses Meta-Tag für die Indizierung verwenden, nur die ersten 20 Wörter indizieren. Stellen Sie sicher, dass Sie die wichtigsten Keywords zuerst eingeben.

Roboter

Das `robots` Attribut, das von mehreren großen Suchmaschinen unterstützt wird, steuert, ob Suchmaschinen-Spider eine Seite indizieren dürfen oder nicht und ob sie Links von einer Seite aus folgen sollen oder nicht.

```
<meta name="robots" content="noindex">
```

In diesem Beispiel werden alle Suchmaschinen angewiesen, die Seite nicht in den Suchergebnissen anzuzeigen. Andere zulässige Werte sind:

Wert / Richtlinie	Bedeutung
`all`	**Standard.** Gleichbedeutend mit `index, follow` . Siehe Anmerkung unten.
`noindex`	Indizieren Sie die Seite überhaupt nicht.
`nofollow`	Folgen Sie nicht den Links auf dieser Seite
`follow`	Die Links auf der Seite können verfolgt werden. Siehe Anmerkung

Wert / Richtlinie	Bedeutung
	unten.
`none`	Entspricht `noindex, nofollow` .
`noarchive`	Machen Sie in den Suchergebnissen keine zwischengespeicherte Version dieser Seite verfügbar.
`nocache`	Synonym für `noarchive` das von einigen Bots wie Bing verwendet wird.
`nosnippet`	Zeigen Sie in den Suchergebnissen keinen Ausschnitt dieser Seite an.
`noodp`	Verwenden Sie keine Metadaten dieser Seite aus dem Open Directory-Projekt für Titel oder Snippets in Suchergebnissen.
`notranslate`	Bieten Sie keine Übersetzungen dieser Seite in den Suchergebnissen an.
`noimageindex`	Bilder auf dieser Seite nicht indexieren.
`unavailable_after [RFC-850 date/time]`	Diese Seite nicht in den Suchergebnissen nach dem angegebenen Datum anzeigen. Datum und Uhrzeit müssen im RFC 850-Format angegeben werden .

Hinweis: Das explizite Definieren des `index` und / oder des `follow` ist nicht erforderlich, solange gültige Werte gelten, da fast alle Suchmaschinen davon ausgehen, dass sie dies tun dürfen, wenn sie nicht ausdrücklich daran gehindert werden. Ähnlich wie bei der Datei "robots.txt" suchen Suchmaschinen im Allgemeinen nur nach Dingen, die sie *nicht ausführen dürfen* . Wenn Sie nur Dinge angeben, die eine Suchmaschine nicht zulässt, dürfen Sie auch nicht versehentlich Gegensätze (wie `index, ..., noindex`) angeben `index, ..., noindex` die nicht alle Suchmaschinen auf dieselbe Weise behandeln.

Rufnummernerkennung

Mobile Plattformen wie iOS erkennen Telefonnummern automatisch und verwandeln sie in `tel:` Links. Während die Funktion sehr praktisch ist, erkennt das System manchmal ISBN-Codes und andere Nummern als Telefonnummern.

Für mobile Safari und andere WebKit-basierte mobile Browser, um die automatische Erkennung und Formatierung von Telefonnummern zu deaktivieren, benötigen Sie dieses Meta-Tag:

```
<meta name="format-detection" content="telephone=no">
```

Sozialen Medien

Open Graph ist ein Standard für Metadaten, der die normalen Informationen erweitert, die in der Kopfmarkierung einer Site enthalten sind. Dadurch können Websites wie Facebook tiefere und reichhaltigere Informationen zu einer Website in strukturierter Form anzeigen. Diese Informationen werden automatisch angezeigt, wenn Benutzer Links zu Websites mit OG-Metadaten auf Facebook freigeben.

Facebook / Diagramm öffnen

```
<meta property="fb:app_id" content="123456789">
<meta property="og:url" content="https://example.com/page.html">
<meta property="og:type" content="website">
<meta property="og:title" content="Content Title">
<meta property="og:image" content="https://example.com/image.jpg">
<meta property="og:description" content="Description Here">
<meta property="og:site_name" content="Site Name">
<meta property="og:locale" content="en_US">
<meta property="article:author" content="">
<!-- Facebook: https://developers.facebook.com/docs/sharing/webmasters#markup -->
<!-- Open Graph: http://ogp.me/ -->
```

- Facebook Open Graph Markup
- Öffnen Sie das Graph-Protokoll

Facebook / Sofortartikel

```
<meta charset="utf-8">
<meta property="op:markup_version" content="v1.0">

<!-- The URL of the web version of your article -->
<link rel="canonical" href="http://example.com/article.html">

<!-- The style to be used for this article -->
<meta property="fb:article_style" content="myarticlestyle">
```

- Facebook Instant-Artikel: Artikel erstellen
- Sofortartikel: Formatreferenz

Twitter verwendet ein eigenes Markup für Metadaten. Diese Metadaten werden als Informationen verwendet, um zu steuern, wie Tweets angezeigt werden, wenn sie einen Link zur Site enthalten.

Twitter

```
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="@site_account">
<meta name="twitter:creator" content="@individual_account">
<meta name="twitter:url" content="https://example.com/page.html">
<meta name="twitter:title" content="Content Title">
<meta name="twitter:description" content="Content description less than 200 characters">
<meta name="twitter:image" content="https://example.com/image.jpg">
```

- Twitter-Karten: Erste Schritte
- Twitter-Kartenprüfer

Google+ / Schema.org

```
<link href="https://plus.google.com/+YourPage" rel="publisher">
<meta itemprop="name" content="Content Title">
<meta itemprop="description" content="Content description less than 200 characters">
<meta itemprop="image" content="https://example.com/image.jpg">
```

Automatische Weiterleitung

Manchmal benötigt Ihre Webseite eine automatische Weiterleitung.

So können Sie beispielsweise nach 5 Sekunden auf `example.com` umleiten:

```
<meta http-equiv="refresh" content="5;url=https://www.example.com/" />
```

Diese Zeile wird Sie an die angegebene Website (in diesem Fall `example.com` nach 5 Sekunden) `example.com` .

Wenn Sie die Zeitverzögerung vor einer Weiterleitung ändern müssen, ändern Sie einfach die Nummer direkt vor Ihrem `;url=` , um die Zeitverzögerung zu ändern.

Web-App

Sie können Ihre Web-App oder Website so einrichten, dass der Startbildschirm eines Geräts ein Anwendungsverknüpfungssymbol hinzugefügt wird. Außerdem kann die App im Vollbildmodus "App-Modus" mit dem Menüelement "Add to Homescreen" von Chrome für Android gestartet werden.

Unter den Meta-Tags werden die Web-Apps im Vollbildmodus (ohne Adressleiste) geöffnet.

Android Chrome

```
<meta name="mobile-web-app-capable" content="yes">
```

IOS

```
<meta name="apple-mobile-web-app-capable" content="yes">
```

Sie können auch die Farbe für Statusleiste und Adressleiste im Meta-Tag festlegen.

Android Chrome

```
<meta name="theme-color" content="black">
```

IOS

```
<meta name="apple-mobile-web-app-status-bar-style" content="black">
```

Meta-Informationen online lesen: https://riptutorial.com/de/html/topic/1264/meta-informationen

Kapitel 32: Navigationsleisten

Examples

Grundlegende Navigationsleiste

Navigationsleisten sind im Wesentlichen eine Liste von Links. Daher werden die Elemente ul und li verwendet, um Navigationslinks einzukapseln.

```
<ul>
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Contact</a></li>
</ul>
```

HTML5-Navigationsleiste

Um eine Navigationsleiste mit dem HTML5-Element nav zu erstellen, hüllen Sie die Links in das nav-Tag ein.

```
<nav>
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Contact</a>
</nav>
```

Navigationsleisten online lesen: https://riptutorial.com/de/html/topic/10725/navigationsleisten

Kapitel 33: Ressourcen verknüpfen

Einführung

Während viele Skripts, Symbole und Stylesheets direkt in HTML-Markup geschrieben werden können, ist es empfehlenswert und effizienter, diese Ressourcen in eine eigene Datei einzubinden und mit Ihrem Dokument zu verknüpfen. In diesem Thema wird das Verknüpfen externer Ressourcen wie Stylesheets und Skripts in ein HTML-Dokument beschrieben.

Syntax

- `<link rel="link-relation" type="mime-type" href="url">`
- `<script src="path-to-script"></script>`

Parameter

Attribut	Einzelheiten
`charset`	Gibt die Zeichenkodierung des verknüpften Dokuments an
`crossorigin`	Gibt an, wie das Element Ursprungsanforderungen behandelt
`href`	Gibt den Speicherort des verknüpften Dokuments an
`hreflang`	Gibt die Sprache des Texts im verknüpften Dokument an
`media`	Gibt an, auf welchem Gerät das verknüpfte Dokument angezeigt wird, und wird häufig verwendet, wenn Stylesheets basierend auf dem betreffenden Gerät ausgewählt werden
`rel`	**Erforderlich** Gibt die Beziehung zwischen dem aktuellen Dokument und dem verknüpften Dokument an
`rev`	Gibt die Beziehung zwischen dem verknüpften Dokument und dem aktuellen Dokument an
`sizes`	Gibt die Größe der verknüpften Ressource an. Nur wenn `rel="icon"`
`target`	Gibt an, wo das verknüpfte Dokument geladen werden soll
`type`	Gibt den Medientyp des verknüpften Dokuments an
`integrity`	Gibt einen Base64-codierten Hash (sha256, sha384 oder sha512) der verknüpften Ressource an, damit der Browser seine Berechtigung überprüfen kann.

Examples

Externes CSS-Stylesheet

```
<link rel="stylesheet" href="path/to.css" type="text/css">
```

Standardmäßig werden CSS-Tags `<link>` im `<head>` -Tag oben in Ihrem HTML-Code platziert. Auf diese Weise wird das CSS zuerst geladen und auf Ihre Seite angewendet, während es geladen wird, anstatt ungestyltes HTML anzuzeigen, bis das CSS geladen wird. Das `type` Attribut ist in HTML5 nicht erforderlich, da HTML5 normalerweise CSS unterstützt.

```
 <link rel="stylesheet" href="path/to.css" type="text/css">
```

und

```
 <link rel="stylesheet" href="path/to.css">
```

... machen Sie dasselbe in HTML5.

Eine andere, wenn auch weniger verbreitete Praxis ist die Verwendung einer `@import` Anweisung in direktem CSS. So was:

```
<style type="text/css">
    @import("path/to.css")
</style>

<style>
    @import("path/to.css")
</style>
```

JavaScript

Synchron

```
<script src="path/to.js"></script>
```

Standardmäßig werden JavaScript-Tags `<script>` unmittelbar vor dem schließenden Tag `</body>` . Wenn Sie Ihre Skripts zuletzt laden, können die visuellen Elemente Ihrer Website schneller angezeigt werden, und Ihr JavaScript wird davon abgehalten, mit Elementen zu interagieren, die noch nicht geladen wurden.

Asynchron

```
<script src="path/to.js" async></script>
```

Wenn der geladene Javascript-Code für die Seiteninitialisierung nicht erforderlich ist, kann er alternativ asynchron geladen werden, wodurch das Laden der Seite beschleunigt wird. Bei Verwendung von `async` lädt der Browser den Inhalt des Skripts parallel und unterbricht nach dem vollständigen Download die HTML-Analyse, um die Javascript-Datei zu analysieren.

Aufgeschoben

```
<script src="path/to.js" defer></script>
```

Zurückgestellte Skripts sind wie asynchrone Skripte, mit der Ausnahme, dass die Analyse nur durchgeführt wird, wenn der HTML-Code vollständig analysiert wurde. Verzögerte Skripte werden garantiert in der Reihenfolge der Deklaration geladen, genau wie synchrone Skripte.

<noscript>

```
<noscript>JavaScript disabled</noscript>
```

Das `<noscript>` -Element definiert den Inhalt, der angezeigt werden soll, wenn der Benutzer Skripts deaktiviert hat oder der Browser die Verwendung von Skripts nicht unterstützt. Das `<noscript>` -Tag kann entweder im `<head>` oder im `<body>` .

Favicon

```
<link rel="icon" type="image/png" href="/favicon.png">
<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico">
```

Verwenden Sie für PNG-Dateien den Mime-Typ `image/png` und für Symboldateien (`*.ico`) das Symbol `image/x-icon` . Für den Unterschied siehe diese SO-Frage .

Eine Datei namens `favicon.ico` im Stammverzeichnis Ihrer Website wird in der Regel automatisch geladen und angewendet, ohne dass ein `<link>` -Tag erforderlich ist. Wenn sich diese Datei ändert, können Browser langsam und hartnäckig sein, um ihren Cache zu aktualisieren.

Alternative CSS

```
<link rel="alternate stylesheet" href="path/to/style.css" title="yourTitle">
```

Bei einigen Browsern können alternative Stylesheets angewendet werden, wenn sie angeboten werden. Standardmäßig werden sie nicht angewendet, können jedoch normalerweise über die Browsereinstellungen geändert werden:

> In Firefox kann der Benutzer das Stylesheet über das Untermenü Ansicht> Seitenstil auswählen. Internet Explorer unterstützt auch diese Funktion (beginnend mit IE 8), auf die auch über Ansicht> Seitenstil (mindestens ab IE 11) zugegriffen werden kann.

Chrome erfordert jedoch eine Erweiterung Verwenden Sie die Funktion (ab Version 48). Die Webseite kann auch eine eigene Benutzeroberfläche bereitstellen, über die der Benutzer die Stile wechseln kann.

(Quelle: die MDN-Dokumente)

Web-Feed

Verwenden Sie das Attribut `rel="alternate"` , um die `rel="alternate"` Ihrer Atom- / RSS-Feeds zu ermöglichen.

```
<link rel="alternate" type="application/atom+xml" href="http://example.com/feed.xml" />
<link rel="alternate" type="application/rss+xml" href="http://example.com/feed.xml" />
```

In den MDN-Dokumenten finden Sie RSS-Feeds und Atomic RSS .

Verknüpfen Sie das 'Medien'-Attribut

```
<link rel="stylesheet" href="test.css" media="print">
```

Medien legt fest, welches Stylesheet für welche Art von Medien verwendet werden soll. Wenn Sie den `print` wird dieses Stylesheet nur für Druckseiten angezeigt.

Der Wert dieses Attributs kann ein beliebiger `mediatype` (ähnlich einer CSS- Medienabfrage).

Zurück und Weiter

Wenn eine Seite eines Teil einer Reihe von Artikeln, zum Beispiel, kann man verwenden `prev` und `next` den Seiten verweisen , die vor und nach kommen.

```
<link rel="prev" href="http://stackoverflow.com/documentation/java/topics">

<link rel="next" href="http://stackoverflow.com/documentation/css/topics">
```

Hinweis zur Ressource: DNS-Prefetch, Prefetch, Prerender

Vorverbindung

Die `preconnect` ähnelt dem DNS `dns-prefetch` `preconnect` dass der DNS aufgelöst wird. Es wird jedoch auch der TCP-Handshake und die optionale TLS-Aushandlung durchgeführt. Dies ist eine experimentelle Funktion.

```
<link rel="preconnect" href="URL">
```

DNS-Prefetch

Informiert Browser über das Auflösen des DNS für eine URL, sodass alle Assets dieser URL schneller geladen werden.

```
<link rel="dns-prefetch" href="URL">
```

Vorabruf

Informiert die Browser darüber, dass eine bestimmte Ressource vorabgerufen werden muss, damit sie schneller geladen werden kann.

```
<link rel="prefetch" href="URL">
```

DNS-Prefetch löst nur den Domänennamen auf, während Prefetch die angegebenen Ressourcen herunterlädt / speichert.

Prerender

Informiert Browser über das Abrufen und Rendern der URL im Hintergrund, sodass sie dem Benutzer sofort übermittelt werden können, wenn der Benutzer zu dieser URL navigiert. Dies ist eine experimentelle Funktion.

```
<link rel="prerender" href="URL">
```

Ressourcen verknüpfen online lesen: https://riptutorial.com/de/html/topic/712/ressourcen-verknupfen

Kapitel 34: Schnittelemente

Bemerkungen

Die HTML5-Standards listen das Hauptelement nicht als Teilelement auf.

Examples

Artikelelement

Das `<article>` -Element enthält **in sich geschlossenen Inhalt** wie Artikel, Blogbeiträge, Benutzerkommentare oder ein interaktives Widget, das außerhalb des Kontexts der Seite, beispielsweise per RSS, verbreitet werden kann.

- Wenn Artikelelemente verschachtelt sind, sollte der Inhalt des inneren Artikelknotens auf das äußere Artikelelement bezogen sein.

Ein Blog (`section`) mit mehreren Beiträgen (`article`) und Kommentaren (`article`) könnte ungefähr so aussehen.

```
<section>
    <!-- Each individual blog post is an <article> -->
    <article>
        <header>
            <h1>Blog Post</h1>
            <time datetime="2016-03-13">13th March 2016</time>
        </header>

        <p>The article element represents a self contained article or document.</p>
        <p>The section element represents a grouping of content.</p>

        <section>
            <h2>Comments <small>relating to "Blog Post"</small></h2>

            <!-- Related comment is also a self-contained article -->
            <article id="user-comment-1">
                <p>Excellent!</p>
                <footer><p>...</p><time>...</time></footer>
            </article>
        </section>
    </article>

    <!-- ./repeat: <article> -->

</section>

<!-- Content unrelated to the blog or posts should be outside the section. -->
<footer>
    <p>This content should be unrelated to the blog.</p>
</footer>
```

Vermeiden Sie unnötigen Gebrauch

Wenn der Hauptinhalt der Seite (mit Ausnahme von Kopfzeilen, Fußzeilen, Navigationsleisten usw.) einfach eine Gruppe von Elementen ist. Sie können den `<article>` zugunsten des `<main>` -Elements weglassen.

```
<article>
    <p>This doesn't make sense, this article has no real `context`.</p>
</article>
```

Stattdessen ersetzen Sie den Artikel mit einem `<main>` Elemente , um anzuzeigen , das den ist `main` für diese Seite.

```
<main>
    <p>I'm the main content, I don't need to belong to an article.</p>
</main>
```

Wenn Sie ein anderes Element verwenden, müssen Sie sicherstellen, dass Sie die ARIA-Rolle `<main>` angeben, um eine korrekte Interpretation und Wiedergabe über mehrere Geräte und Nicht-HTML5-Browser hinweg zu ermöglichen.

```
<section role="main">
    <p>This section is the main content of this page.</p>
</section>
```

Anmerkungen:

- Nachkommen von `<main>` -Elementen sind in einem `<article>` nicht zulässig

Klicken Sie hier, um die offizielle HTML5-Spezifikation für das `<article>` -Element zu lesen

Hauptelement

Das `<main>` -Element enthält den **Hauptinhalt** für Ihre Webseite. Dieser Inhalt ist für die einzelne Seite eindeutig und sollte an keiner anderen Stelle der Website angezeigt werden. Wiederholte Inhalte wie Kopf- und Fußzeilen, Navigation, Logos usw. werden außerhalb des Elements platziert.

- Das `<main>` -Element sollte auf einer Seite höchstens **einmal verwendet werden** .
- Das `<main>` -Element darf nicht als Nachkomme eines `article` , `aside` `footer` , `header` oder `nav` eingefügt werden.

Im folgenden Beispiel zeigen wir einen **einzelnen Blogeintrag** (und zugehörige Informationen wie Referenzen und Kommentare).

```
<body>
    <header>
```

```
        <nav>...</nav>
    </header>

    <main>
        <h1>Individual Blog Post</h1>
        <p>An introduction for the post.</p>

        <article>
            <h2>References</h2>
            <p>...</p>
        </article>

        <article>
            <h2>Comments</h2> ...
        </article>
    </main>

    <footer>...</footer>
</body>
```

- Der Blogbeitrag ist im Element `<main>` , um darauf hinzuweisen, dass dies der Hauptinhalt dieser Seite ist (und somit eindeutig auf der gesamten Website).
- Die `<header>` und `<footer>` Tags sind *Geschwister* mit dem `<main>` Element.

Anmerkungen:

Die HTML5-Spezifikation erkennt das `<main>` -Element als **Gruppierungselement** und nicht als ein *Abschnittselement* .

- ARIA-Rollenattribute : `main` *(Standard)* , `presentation`

Das Hinzufügen eines `role="main"` ARIA- Rollenattributs zu **anderen Elementen** , die als Hauptinhalt verwendet werden sollen, wird empfohlen, um Benutzeragenten zu unterstützen, die kein HTML5 unterstützen, und um mehr Kontext für diejenigen bereitzustellen, die dies tun.

Das `<main>` -Element hat standardmäßig die Hauptrolle und muss daher nicht angegeben werden.

Klicken Sie hier, um die offizielle HTML5-Spezifikation für das `<main>` -Element zu lesen

Nav-Element

Das Element `<nav>` ist in erster Linie für Abschnitte gedacht, die **Hauptnavigationsblöcke** für die Website enthalten. Dies kann Links zu anderen Teilen der Website *(z. B. Anker für ein Inhaltsverzeichnis)* oder andere Seiten enthalten.

Inline-Artikel

Im Folgenden werden eine Reihe von Hyperlinks angezeigt.

```
<nav>
    <a href="https://google.com">Google</a>
    <a href="https://www.yahoo.com">Yahoo!</a>
    <a href="https://www.bing.com">Bing</a>
</nav>
```

Verwenden Sie bei Bedarf Listenelemente

Wenn der Inhalt eine Liste von Elementen darstellt, verwenden Sie ein Listenelement, um dies anzuzeigen und die Benutzererfahrung zu verbessern.

Beachten Sie die `role="navigation"` , dazu *weiter unten.*

```
<nav role="navigation">
    <ul>
        <li><a href="https://google.com">Google</a></li>
        <li><a href="https://www.yahoo.com">Yahoo!</a></li>
        <li><a href="https://www.bing.com">Bing</a></li>
    </ul>
</nav>
```

Vermeiden Sie unnötigen Gebrauch

`<footer>` -Elemente enthalten möglicherweise eine Liste mit Links zu anderen Teilen der Website (FAQ, AGB usw.). Das Fußzeile Element allein ist in diesem Fall ausreichend, *brauchen* Sie nicht weiter zu Ihren Links mit einem Wrap `<nav>` Elemente in dem `<footer>` .

```
<!-- the <nav> is not required in the <footer> -->
<footer>
    <nav>
        <a href="#">...</a>
    </nav>
</footer>

<!-- The footer alone is sufficient -->
<footer>
    <a href="#">...</a>
</footer>
```

Anmerkungen:

- Nachkommen von `<main>` -Elementen sind in einem `<nav>` nicht zulässig

Wenn Sie dem `<nav>` -Element eine ARIA-Rolle `role="navigation"` `<nav>` , wird empfohlen, Benutzeragenten zu unterstützen, die HTML5 nicht unterstützen, und denjenigen, die dies tun, mehr Kontext bereitzustellen.

```
<nav role="navigation"><!-- ... --></nav>
```

Bildschirmleser: *(Software, mit der blinde oder sehbehinderte Benutzer auf der Website navigieren können)*

Benutzeragenten wie Screenreader interpretieren das `<nav>` -Element je nach ihren Anforderungen unterschiedlich.

- Es könnte dem `<nav>` -Element beim Rendern der Seite eine höhere Priorität einräumen
- Dies kann das Rendern des Elements verzögern
- Es könnte die Seite auf eine bestimmte Weise anpassen, um sie an die Bedürfnisse des Benutzers anzupassen
  Beispiel: Vergrößern Sie die Textlinks innerhalb der `<nav>` -Elemente für sehbehinderte Personen.

Klicken Sie hier, um die offizielle HTML5-Spezifikation für das `<nav>` -Element zu lesen

Abschnittselement

Das `<section>` -Element repräsentiert einen generischen Abschnitt, um Inhalte thematisch zu gruppieren. In der Regel sollte jeder Abschnitt mit einem Überschriftenelement als untergeordnetes Element des `section` identifiziert werden können.

- Sie können das `<section>` -Element innerhalb eines `<article>` und umgekehrt.
- Jeder Abschnitt sollte ein *Thema haben* (ein Überschriftenelement, das diese Region identifiziert).
- Verwenden Sie das `<section>` -Element nicht als allgemeinen "Container".
  Wenn Sie einen Container zum Anwenden des `<div>` benötigen, verwenden Sie stattdessen ein `<div>` .

Im folgenden Beispiel zeigen wir einen **einzelnen Blogbeitrag** mit mehreren Kapiteln. Jedes Kapitel ist ein Abschnitt *(eine Gruppe thematisch gruppierter Inhalte, die durch die Überschriftenelemente in jedem Abschnitt identifiziert werden können)* .

```
<article>
    <header>
        <h2>Blog Post</h2>
    </header>
    <p>An introduction for the post.</p>
    <section>
        <h3>Chapter 1</h3>
        <p>...</p>
    </section>
    <section>
        <h3>Chapter 2</h3>
        <p>...</p>
    </section>
    <section>
        <h3>Comments</h3> ...
    </section>
</article>
```

Anmerkungen:

Entwickler sollten das **article-** Element verwenden, wenn es sinnvoll ist, den Inhalt des Elements zu syndizieren.

Klicken Sie hier, um die offizielle HTML5-Spezifikation für das `<main>` -Element zu lesen

Kopfelement

Das `<header>` -Element repräsentiert den einleitenden Inhalt für seinen nächsten Abschnittsinhalt oder Abschnittsstammelement. Ein `<header>` enthält normalerweise eine Gruppe von Einführungs- oder Navigationshilfen.

Hinweis: Das Header-Element ist nicht in Abschnitte unterteilt. Es wird kein neuer Abschnitt eingeführt.

Beispiele:

```
<header>
  <p>Welcome to...</p>
  <h1>Voidwars!</h1>
</header>
```

In diesem Beispiel hat der `<article>` einen `<header>` .

```
<article>
  <header>
    <h1>Flexbox: The definitive guide</h1>
  </header>
  <p>The guide about Flexbox was supposed to be here, but it turned out Wes wasn’t a Flexbox
expert either.</p>
</article>
```

Vorgeschlagene Empfehlung des W3C

Fußzeilenelement

Das `<footer>` -Element enthält den Fußzeilenteil der Seite.

Hier ist ein Beispiel für das `<footer>` -Element, das das `p` Absatztag enthält

```
<footer>
    <p>All rights reserved</p>
</footer>
```

Schnittelemente online lesen: https://riptutorial.com/de/html/topic/311/schnittelemente

Kapitel 35: Segeltuch

Parameter

Attribut	Beschreibung
Höhe	Gibt die Leinwandhöhe an
Breite	Gibt die Leinwandbreite an

Bemerkungen

- Dieses Tag ist nicht mit weniger als 9 Versionen von Internet Explorer kompatibel. Überprüfen Sie caniuse.com auf Browserkompatibilität.
- `canvas` ist nur ein Container für Grafiken, und das eigentliche Zeichnen von Grafiken wird von JavaScript ausgeführt.

Examples

Basisbeispiel

Das `canvas` Element wurde in HTML5 zum Zeichnen von Grafiken eingeführt.

```
<canvas id="myCanvas">
   Cannot display graphic. Canvas is not supported by your browser (IE<9)
</canvas>
```

Das Obige erstellt ein transparentes HTML `<canvas>` -Element mit einer Größe von 300 × 150 px.

Mit dem **Canvas-** Element können Sie erstaunliche Elemente wie Formen und Grafiken zeichnen, Bilder bearbeiten, ansprechende Spiele erstellen usw. mit **JavaScript** .
Die `canvas` ist 2D *ziehbar Schichtoberfläche* Objekt wird bezeichnet als `CanvasRenderingContext2D` ; oder aus einem `HTMLCanvasElement` mit der `.getContext("2d")` Methode:

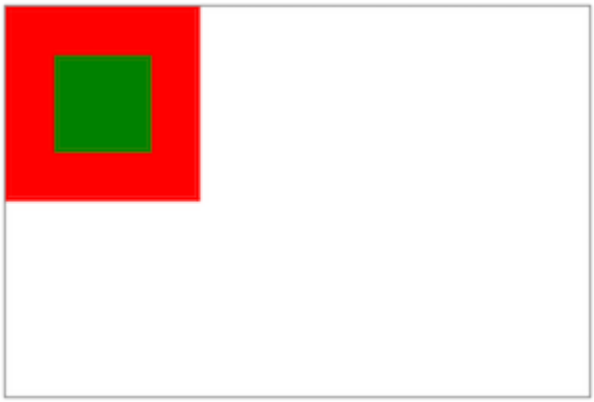
```
var ctx = document.getElementById("myCanvas").getContext("2d");
// now we can refer to the canvas's 2D layer context using `ctx`

ctx.fillStyle = "#f00";
ctx.fillRect(0, 0, ctx.canvas.width, ctx.canvas.height); // x, y, width, height

ctx.fillStyle = "#000";
ctx.fillText("My red canvas with some black text", 24, 32); // text, x, y
```

jsFiddle-Beispiel

Zeichnen von zwei Rechtecken auf einem

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>Draw two rectangles on the canvas</title>
    <style>
      canvas{
          border:1px solid gray;
      }
    </style>
    <script async>
      window.onload = init; // call init() once the window is completely loaded
      function init(){
        // #1 - get reference to <canvas> element
        var canvas = document.querySelector('canvas');

        // #2 - get reference to the drawing context and drawing API
        var ctx = canvas.getContext('2d');

        // #3 - all fill operations are now in red
        ctx.fillStyle = 'red';

        // #4 - fill a 100x100 rectangle at x=0,y=0
        ctx.fillRect(0,0,100,100);

        // #5 - all fill operations are now in green
        ctx.fillStyle = 'green';

        // #6 - fill a 50x50 rectangle at x=25,y=25
        ctx.fillRect(25,25,50,50);

      }
      </script>
</head>
<body>
  <canvas width=300 height=200>Your browser does not support canvas.</canvas>
</body>
</html>
```

Dieses Beispiel sieht folgendermaßen aus:



Segeltuch online lesen: https://riptutorial.com/de/html/topic/1162/segeltuch

Kapitel 36: SVG

Einführung

SVG steht für Scalable Vector Graphics. SVG wird verwendet, um Grafiken für das Web zu definieren

Das HTML-Element `<svg>` ist ein Container für SVG-Grafiken.

SVG bietet mehrere Methoden zum Zeichnen von Pfaden, Boxen, Kreisen, Text und Grafiken.

Bemerkungen

SVG ist eine XML-basierte Sprache zum Erstellen skalierbarer Vektorbilder. Es kann direkt in ein HTML-Dokument geschrieben oder aus externen SVG-Dateien eingebettet werden. Inline-SVG kann mit CSS bzw. JavaScript neu gestaltet und geändert werden.

Die Browser-Unterstützung für SVG ist unterschiedlich, kann hier jedoch ermittelt werden .

Ausführlichere Informationen finden Sie in der SVG-Dokumentation .

Examples

Einbetten externer SVG-Dateien in HTML

Sie können die Elemente `<img>` oder `<object>` verwenden, um externe SVG-Elemente einzubetten. Die Einstellung der Höhe und Breite ist optional, wird jedoch dringend empfohlen.

Verwenden des Bildelements

```
<img src="attention.svg" width="50" height="50">
```

Mit `<img>` können Sie die SVG- `<img>` nicht mit CSS formatieren oder mit JavaScript bearbeiten.

Verwenden des Objektelements

```
<object type="image/svg+xml" data="attention.svg" width="50" height="50">
```

Im Gegensatz zu `<img>` importiert `<object>` das SVG direkt in das Dokument und kann daher mit Javascript und CSS bearbeitet werden.

Inline-SVG

SVG kann direkt in ein HTML-Dokument geschrieben werden. Inline-SVG kann mit CSS und

JavaScript gestaltet und bearbeitet werden.

```
<body>
    <svg class="attention" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1000 1000" >
        <path id="attention" d="m571,767l0,-106q0,-8,-5,-13t-12,-5l-108,0q-7,0,-12,5t-
5,13l0,106q0,8,5,13t12,6l108,0q7,0,12,-6t5,-13Zm-1,-208l10,-257q0,-6,-5,-10q-7,-6,-14,-6l-
122,0q-7,0,-14,6q-5,4,-5,12l9,255q0,5,6,9t13,3l103,0q8,0,13,-3t6,-9Zm-7,-522l428,786q20,35,-
1,70q-10,17,-26,26t-35,10l-858,0q-18,0,-35,-10t-26,-26q-21,-35,-1,-70l429,-786q9,-17,26,-
27t36,-10t36,10t27,27Z" />
    </svg>
</body>
```

Das obige Inline-SVG kann dann mit der entsprechenden CSS-Klasse gestaltet werden:

```
.attention {
    fill: red;
    width: 50px;
    height: 50px;
}
```

Das Ergebnis sieht so aus:

Einbetten von SVG mit CSS

Sie können externe SVG-Dateien mit der Eigenschaft `background-image` hinzufügen, genau wie bei jedem anderen Bild.

HTML:

```
<div class="attention"></div>
```

CSS:

```
.attention {
    background-image: url(attention.svg);
    background-size: 100% 100%;
    width: 50px;
    height: 50px;
}
```

Sie können das Bild auch direkt in eine CSS-Datei mit einer Daten-URL einbetten:

```
background-image:
url(data:image/svg+xml,%3Csvg%20xmlns%3D%22http%3A%2F%2Fwww.w3.org%2F2000%2Fsvg%22%20xmlns%3Axlink%3D%2
106q0%2C-8%2C-5%2C-13t-12%2C-5l-108%2C0q-7%2C0%2C-12%2C5t-
5%2C13l0%2C106q0%2C8%2C5%2C13t12%2C6l108%2C0q7%2C0%2C12%2C-6t5%2C-13Zm-1%2C-208l10%2C-
257q0%2C-6%2C-5%2C-10q-7%2C-6%2C-14%2C-6l-122%2C0q-7%2C0%2C-14%2C6q-5%2C4%2C-
5%2C12l9%2C255q0%2C5%2C6%2C9t13%2C3l103%2C0q8%2C0%2C13%2C-3t6%2C-9Zm-7%2C-
```

```
5221428%2C786q20%2C35%2C-1%2C70q-10%2C17%2C-26%2C26t-35%2C10l-858%2C0q-18%2C0%2C-35%2C-10t-
26%2C-26q-21%2C-35%2C-1%2C-701429%2C-786q9%2C-17%2C26%2C-27t36%2C-
10t36%2C10t27%2C27Z%22%20%2F%3E%0D%0A%3C%2Fsvg%3E);
```

SVG online lesen: https://riptutorial.com/de/html/topic/1183/svg

Kapitel 37: Tabellen

Einführung

Mit dem HTML-Element `<table>` können Web-Autoren Tabellendaten (z. B. Text, Bilder, Links, andere Tabellen usw.) in einer zweidimensionalen Tabelle mit Zeilen und Spalten von Zellen anzeigen.

Syntax

- `<table></table>`
- `<thead></thead>`
- `<tbody></tbody>`
- `<tfoot></tfoot>`
- `<tr></tr>`
- `<th></th>`
- `<td></td>`

Bemerkungen

Die verschiedenen Tabellenelemente und ihre Inhaltsattribute definieren zusammen das Tabellenmodell. Das `<table>` -Element ist das Containerelement für Tabellenmodelle / Tabellendaten. Tabellen haben Zeilen, Spalten und Zellen, die von ihren Nachkommen angegeben werden. Die Zeilen und Spalten bilden ein Gitter. Die Zellen einer Tabelle müssen dieses Raster vollständig überdecken, ohne sich zu überlappen. Die folgende Liste beschreibt die verschiedenen Elemente des Tabellenmodells:

- `<table>` - Das Containerelement für Tabellenmodelle / Tabellendaten. `<table>` repräsentiert Daten mit mehr als einer Dimension in Form einer Tabelle.
- `<caption>` - Untertitel oder Titel der Tabelle (wie eine `figcaption` einer `figure`)
- `<col>` - Eine Spalte (ein Element ohne Inhalt)
- `<colgroup>` - Eine Gruppierung von Spalten
- `<thead>` - Tabellenkopf (nur eine)
- `<tbody>` - Tabellenkörper / Inhalt (mehrere sind in Ordnung)
- `<tfoot>` - Tabellenfußzeile (nur eine)
- `<tr>` - Tabellenzeile
- `<th>` - Tabellenkopfzelle
- `<td>` - Tabellenzelle

Semantisch sind Tabellen für das Halten von Tabellendaten gedacht. Sie können es sich als eine Möglichkeit vorstellen, Daten anzuzeigen und zu beschreiben, die in einem Arbeitsblatt sinnvoll sind (Spalten und Zeilen).

Die Verwendung von Tabellen für das Layout wird nicht empfohlen. Verwenden Sie stattdessen CSS-Regeln für Layout und Formatierung, einschließlich `display: table` .

Eine bemerkenswerte Ausnahme, die in der Branche in Bezug auf die Verwendung des `<table>` -Layouts angezeigt wird, betrifft HTML-E-Mail: Einige E-Mail-Clients, einschließlich Outlook, wurden auf ältere Rendering-Engines zurückgesetzt, nachdem Microsoft seinen Monopolfall im Vergleich zur EU verloren hatte. Damit Microsoft den IE nicht in das Betriebssystem integriert, wurde das Rendering-Modul von Outlook auf eine frühere Version von Trident zurückgesetzt. Dieses Rollback unterstützt moderne Web-Technologien einfach nicht. Daher ist die Verwendung von `<table>` -basierten Layouts für HTML-E-Mail die einzige Möglichkeit, die Browser- / Plattform / Client-Kompatibilität zu gewährleisten.

Examples

Einfache Tabelle

```
<table>
  <tr>
    <th>Heading 1/Column 1</th>
    <th>Heading 2/Column 2</th>
  </tr>
  <tr>
    <td>Row 1 Data Column 1</td>
    <td>Row 1 Data Column 2</td>
  </tr>
  <tr>
    <td>Row 2 Data Column 1</td>
    <td>Row 2 Data Column 2</td>
  </tr>
</table>
```

Dadurch wird eine `<table>` die aus drei Zeilen (`<tr>`) besteht: einer Zeile mit Kopfzeilen (`<th>`) und zwei Zeilen mit Inhaltszellen (`<td>`). `<th>` -Elemente sind *tabellarische Überschriften* und `<td>` -Elemente *tabellarische Daten* . Sie können alles, was Sie möchten, in ein `<td>` oder `<th>` .

Überschrift 1 / Spalte 1	Überschrift 2 / Spalte 2
Zeile 1 Datenspalte 1	Zeile 1 Datenspalte 2
Zeile 2 Datenspalte 1	Zeile 2 Datenspalte 2

Spalten oder Zeilen überspannen

Tabellenzellen können sich über mehrere Spalten oder Zeilen erstrecken, wobei die Attribute `colspan` und `rowspan` werden. Diese Attribute können auf die Elemente `<th>` und `<td>` angewendet werden.

```
<table>
    <tr>
        <td>row 1 col 1</td>
        <td>row 1 col 2</td>
        <td>row 1 col 3</td>
    </tr>
```

```
    <tr>
        <td colspan="3">This second row spans all three columns</td>
    </tr>
    <tr>
        <td rowspan="2">This cell spans two rows</td>
        <td>row 3 col 2</td>
        <td>row 3 col 3</td>
    </tr>
    <tr>
        <td>row 4 col 2</td>
        <td>row 4 col 3</td>
    </tr>

</table>
```

Wird darin enden, dass

row 1 col 1	row 1 col 2	row 1 col 3
This second row spans all three columns		
This cell spans two rows	row 3 col 2	row 3 col 3
	row 4 col 2	row 4 col 3

Beachten Sie, dass Sie keine Tabelle entwerfen sollten, bei der sich Zeilen und Spalten überlappen, da dies ungültiges HTML ist und das Ergebnis von verschiedenen Webbrowsern unterschiedlich behandelt wird.

`rowspan` = Eine nicht negative ganze Zahl, die die Anzahl der Zeilen angibt, die von einer Zelle überspannt werden. Der Standardwert dieses Attributs ist eins (`1`). Ein Wert von null (`0`) bedeutet, dass sich die Zelle von der aktuellen Zeile bis zur letzten Zeile der Tabelle erstreckt (`<thead>` , `<tbody>` oder `<tfoot>`).

`colspan` = Eine nicht negative ganze Zahl, die die Anzahl der Spalten angibt, die von der aktuellen Zelle überspannt werden. Der Standardwert dieses Attributs ist eins (`1`). Ein Wert von null (`0`) bedeutet, dass sich die Zelle von der aktuellen Spalte bis zur letzten Spalte der Spaltengruppe `<colgroup>` in der die Zelle definiert ist.

Tisch mit Kopf, Fuß, Fuß und Bildunterschrift

HTML stellt den Tabellen auch die `<thead>` , `<tbody>` , `<tfoot>` und `<caption>` . Diese zusätzlichen Elemente sind nützlich, um Ihren Tabellen einen semantischen Wert hinzuzufügen und Platz für ein separates CSS-Styling zu bieten.

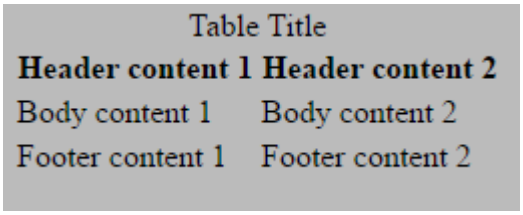
Beim Ausdrucken einer Tabelle, die nicht auf eine (Papierseite) passt, wird bei den meisten Browsern der Inhalt von `<thead>` auf jeder Seite wiederholt.

Es gibt eine bestimmte Reihenfolge, die eingehalten werden muss, und wir sollten uns bewusst sein, dass nicht jedes Element wie erwartet in Einklang kommt. Das folgende Beispiel zeigt, wie unsere 4 Elemente platziert werden sollten.

```
<table>
```

```
 <caption>Table Title</caption> <!--| caption is the first child of table |-->
  <thead> <!--======================| thead is after caption |-->
    <tr>
      <th>Header content 1</th>
      <th>Header content 2</th>
    </tr>
  </thead>

  <tbody> <!--======================| tbody is after thead |-->
    <tr>
      <td>Body content 1</td>
      <td>Body content 2</td>
    </tr>
  </tbody>

  <tfoot><!--| tfoot can be placed before or after tbody, but not in a group of tbody. |-->
<!--| Regardless where tfoot is in markup, it's rendered at the bottom. |-->

    <tr>
      <td>Footer content 1</td>
      <td>Footer content 2</td>
    </tr>
  </tfoot>

</table>
```

Im folgenden Beispiel werden die Ergebnisse werden zweimal gezeigt - die erste Tabelle fehlt alle Stile, die zweite Tabelle hat ein paar CSS - Eigenschaften angewendet: `background-color` , `color` und `border` *. Die Stile werden als visueller Leitfaden bereitgestellt und sind kein wesentlicher Aspekt des vorliegenden Themas.



Element	Stile trifft zu
`<caption>`	Gelber Text auf schwarzem Hintergrund.
`<thead>`	Mutiger Text auf purpurrotem Hintergrund.
`<tbody>`	Text auf blauem Hintergrund.
`<tfoot>`	Text auf grünem Hintergrund.

Element	Stile trifft zu
`<th>`	Orange Grenzen.
`<td>`	Rote Grenzen

Spaltengruppen

In manchen Fällen möchten Sie möglicherweise eine Spalte oder eine Gruppe von Spalten formatieren. Oder Sie möchten aus semantischen Gründen Spalten zusammenfassen. Verwenden `<colgroup>` Elemente `<colgroup>` und `<col>` .

Mit dem optionalen `<colgroup>` -Tag können Sie Spalten zusammenfassen. `<colgroup>` -Elemente müssen `<colgroup>` Elemente einer `<table>` und müssen hinter jedem `<caption>` -Element und vor jedem Tabelleninhalt (z. B. `<tr>` , `<thead>` , `<tbody>` usw.) stehen.

```
<table>
    <colgroup span="2"></colgroup>
    <colgroup span="2"></colgroup>
    ...
</table>
```

Mit dem optionalen `<col>` -Tag können Sie auf einzelne Spalten oder einen Spaltenbereich verweisen, ohne eine logische Gruppierung anzuwenden. `<col>` -Elemente sind optional, aber falls vorhanden, müssen sie sich innerhalb eines `<colgroup>` -Elements befinden.

```
<table>
    <colgroup>
        <col id="MySpecialColumn" />
        <col />
    </colgroup>
    <colgroup>
        <col class="CoolColumn" />
        <col class="NeatColumn" span="2" />
    </colgroup>
    ...
</table>
```

Die folgenden CSS-Stile können auf die Elemente `<colgroup>` und `<col>` angewendet werden:

- `border`
- `background`
- `width`
- `visibility`
- `display` (wie in `display: none`)
    - `display: none;` entfernt tatsächlich die Spalten aus der Anzeige, wodurch die Tabelle so gerendert wird, als wären diese Zellen nicht vorhanden

Weitere Informationen finden Sie unter HTML5-Tabellendaten .

Überschrift

`th` Elemente werden sehr häufig verwendet, um Überschriften für Tabellenzeilen und -spalten anzuzeigen, wie folgt:

```
<table>
    <thead>
        <tr>
            <td></td>
            <th>Column Heading 1</th>
            <th>Column Heading 2</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <th>Row Heading 1</th>
            <td></td>
            <td></td>
        </tr>
        <tr>
            <th>Row Heading 2</th>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

Dies kann durch die Verwendung des `scope` für die Zugänglichkeit verbessert werden. Das obige Beispiel würde wie folgt geändert:

```
<table>
    <thead>
        <tr>
            <td></td>
            <th scope="col">Column Heading 1</th>
            <th scope="col">Column Heading 2</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <th scope="row">Row Heading 1</th>
            <td></td>
            <td></td>
        </tr>
        <tr>
            <th scope="row">Row Heading 1</th>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

`scope` wird als *Aufzählungsattribut bezeichnet* , was bedeutet, dass es einen Wert aus einem bestimmten Satz möglicher Werte haben kann. Dieses Set beinhaltet:

- `col`
- `row`
- `colgroup`
- `rowgroup`

Verweise:

- https://developer.mozilla.org/en-US/docs/Web/HTML/Element/th#attr-scope
- https://www.w3.org/TR/WCAG20-TECHS/H63.html

Tabellen online lesen: https://riptutorial.com/de/html/topic/274/tabellen

Kapitel 38: Tabindex

Parameter

Wert	Bedeutung
Negativ	Das Element wird fokussierbar sein, es sollte jedoch nicht über die sequentielle Tastaturnavigation erreichbar sein
0	Das Element kann über die sequentielle Tastaturnavigation fokussiert und erreicht werden, seine relative Reihenfolge wird jedoch durch die Plattformkonvention festgelegt
positiv	Das Element muss fokussierbar und über die sequentielle Tastaturnavigation zugänglich sein. Die relative Reihenfolge wird durch den Attributwert definiert: Das sequentielle folgt der aufsteigenden Nummer des `tabindex`

Bemerkungen

Der Höchstwert für `tabindex` sollte 32767 gemäß W3C-Abschnitt 17.11.1 nicht überschreiten, sofern der Standardwert nicht -1 ist

Ein Element mit dem Wert 0, einem ungültigen Wert oder keinem `tabindex` Wert sollte nach den Elementen mit einem positiven Index in der Reihenfolge der Tastaturnavigation angeordnet werden.

Examples

Fügen Sie der Tab-Reihenfolge ein Element hinzu

```
<div tabindex="0">Some button</div>
```

Hinweis: Versuchen Sie eine native HTML verwenden `button` oder einen `a` Tag , wo angemessen.

Entfernen Sie ein Element aus der Tab-Reihenfolge

```
<button tabindex="-1">This button will not be reachable by tab</button>
```

Das Element wird aus der Tab-Reihenfolge entfernt, kann jedoch weiterhin fokussiert werden.

Definieren einer benutzerdefinierten Tab-Reihenfolge (nicht empfohlen)

```
<div tabindex="2">Second</div>
```

```
<div tabindex="1">First</div>
```

Bei positiven Werten wird das Element an der Tabulatorreihenfolge des entsprechenden Werts eingefügt. Elemente ohne Präferenz (dh `tabindex="0"` oder native Elemente wie `button` und `a` werden nach denen mit Präferenz angehängt.

Positive Werte werden **nicht empfohlen,** da sie das erwartete Tabbing-Verhalten stören und möglicherweise Personen verwirren, die auf Screenreader angewiesen sind. Versuchen Sie, eine natürliche Reihenfolge zu erstellen, indem Sie Ihre DOM-Struktur neu anordnen.

Tabindex online lesen: https://riptutorial.com/de/html/topic/2594/tabindex

Kapitel 39: Textformatierung

Einführung

Während die meisten HTML-Tags zum Erstellen von Elementen verwendet werden, enthält HTML auch In-Text-Formatierungs-Tags, um bestimmte textbezogene Stile auf Textabschnitte anzuwenden. Dieses Thema enthält Beispiele für die HTML-Textformatierung, z. B. Hervorheben, Fettdruck, Unterstreichungen, Index und Text.

Syntax

- `<abbr>Abbreviation</abbr>`
- `<b>Bold Text</b>`
- `<del>Deleted Text</del>`
- `<em>Emphasized Text</em>`
- `<i>Italic Text</i>`
- `<ins>Inserted Text</ins>`
- `<mark>Marked (or Highlighted) Text</mark>`
- `<s>Stricken Text</s>`
- `<strong>Strong Text</strong>`
- `<sub>Subscript Text</sub>`
- `<sup>Superscript Text</sup>`
- `<u>Underlined Text</u>`

Examples

Fett, Kursiv und Unterstrichen

Fett Text

Verwenden Sie zum Markieren von Text die Tags `<strong>` oder `<b>` :

```
<strong>Bold Text Here</strong>
```

oder

```
<b>Bold Text Here</b>
```

Was ist der Unterschied? Semantik. `<strong>` wird verwendet, um anzuzeigen, dass der Text für den umgebenden Text von grundlegender oder semantischer *Bedeutung ist* , während `<b>` keine solche Bedeutung angibt und lediglich Text darstellt, der fett gedruckt werden soll.

Wenn Sie `<b>` ein Text-to-Speech-Programm die Wörter nicht anders als die anderen Wörter in der Umgebung ausdrücken. Sie lenken die Aufmerksamkeit einfach auf sie, ohne zusätzliche Bedeutung hinzuzufügen. Wenn Sie jedoch `<strong>` , möchte das gleiche Programm diese Wörter mit einem anderen Ton sprechen, um zu vermitteln, dass der Text in irgendeiner Weise wichtig ist.

Kursiver Text

Verwenden Sie zum Kursivieren von Text die Tags `<em>` oder `<i>` :

```
<em>Italicized Text Here</em>
```

oder

```
<i>Italicized Text Here</i>
```

Was ist der Unterschied? Semantik. `<em>` wird verwendet, um anzugeben, dass der Text eine zusätzliche Betonung haben sollte, die hervorgehoben werden sollte, während `<i>` nur Text darstellt, der vom normalen Text um ihn herum entfernt werden sollte.

Wenn Sie zum Beispiel die Aktion innerhalb eines Satzes betonen möchten, können Sie dies durch `<em>` kursiv hervorheben: "Möchten Sie die Änderung bereits *einreichen* ?"

Wenn Sie jedoch ein Buch oder eine Zeitung identifizieren würden, die Sie normalerweise stilistisch kursiv darstellen würden, verwenden Sie einfach `<i>` : "Ich musste *Romeo und Julia* in der High School lesen.

Unterstrichener Text

Während das `<u>` -Element selbst in HTMl 4 nicht mehr empfohlen wird, wurde es mit alternativer semantischer Bedeutung in HTML 5 wieder eingeführt - um eine nicht artikulierte, nichttextuelle Anmerkung darzustellen. Sie können ein solches Rendering verwenden, um falsch geschriebenen Text auf der Seite anzuzeigen oder eine chinesische Eigenmarke zu verwenden.

```
<p>This paragraph contains some <u>mispelled</u> text.</p>
```

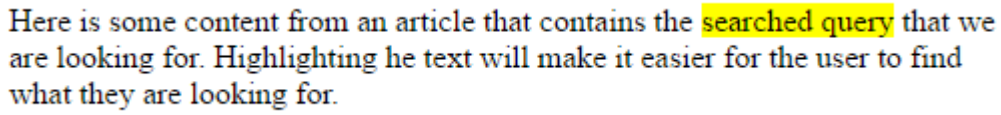
Hervorhebung

Das `<mark>` -Element ist neu in HTML5 und wird verwendet, um Text in einem Dokument "aufgrund seiner Relevanz in einem anderen Kontext" zu markieren oder hervorzuheben. [1]

Das üblichste Beispiel wäre in den Ergebnissen einer Suche, wenn der Benutzer eine Suchanfrage eingegeben hat und die Ergebnisse hervorgehoben werden, die die gewünschte Abfrage hervorheben.

```
<p>Here is some content from an article that contains the <mark>searched query</mark>
that we are looking for. Highlighting the text will make it easier for the user to
find what they are looking for.</p>
```

Ausgabe:



Eine gängige Standardformatierung ist schwarzer Text auf gelbem Hintergrund. Dies kann jedoch mit CSS geändert werden.

Eingefügt, gelöscht oder gestrichen

Um Text als eingefügt zu markieren, verwenden Sie das `<ins>` -Tag:

```
<ins>New Text</ins>
```

Um Text als gelöscht zu markieren, verwenden Sie das `<del>` -Tag:

```
<del>Deleted Text</del>
```

Um den Text durchzublättern, verwenden Sie das `<s>` -Tag:

```
<s>Struck-through text here</s>
```

Hochgestellt und tiefgestellt

Um Text nach oben oder unten zu verschieben, können Sie die Tags `<sup>` und `<sub>` .

So erstellen Sie hochgestellt:

```
<sup>superscript here</sup>
```

So erstellen Sie ein Index:

```
<sub>subscript here</sub>
```

Abkürzung

Um einen Ausdruck als Abkürzung zu kennzeichnen, verwenden Sie das `<abbr>` -Tag:

```
<p>I like to write <abbr title="Hypertext Markup Language">HTML</abbr>!</p>
```

Wenn vorhanden, wird das `title` Attribut verwendet, um die vollständige Beschreibung dieser Abkürzung darzustellen.

Textformatierung online lesen: https://riptutorial.com/de/html/topic/526/textformatierung

Kapitel 40: Überschriften

Einführung

HTML bietet nicht nur einfache Absatz-Tags, sondern sechs separate Header-Tags, um Überschriften verschiedener Größen und Stärken anzuzeigen. Überschrift 1 bis Überschrift 6 enthält die größten und dicksten Texte, während Überschrift 6 bis zur Absatzebene der kleinste und dünnste ist. In diesem Thema wird die ordnungsgemäße Verwendung dieser Tags beschrieben.

Syntax

- `<h1>...</h1>`
- `<h2>...</h2>`
- `<h3>...</h3>`
- `<h4>...</h4>`
- `<h5>...</h5>`
- `<h6>...</h6>`

Bemerkungen

- Ein `h1` - `h6` Element muss sowohl ein Starttag als auch ein Endtag haben. [1]
- `h1` - `h6` Elemente sind standardmäßig Elemente auf Blockebene (CSS-Stil: `display: block`). [2]
- `h1` - `h6` Elemente sollten nicht mit dem Abschnittselement verwechselt werden
- Überschriften-Tags (`h1` - `h6`) beziehen sich nicht auf das `head` Tag.
- Zulässiger Inhalt: Formulierungsinhalt
- Die verschiedenen CSS-Stile für Überschriften unterscheiden sich normalerweise in der `font-size` und im `margin` . Die folgenden CSS-Einstellungen für `h1` - `h6` Elemente können als Orientierung dienen (vom W3C als "informativ" bezeichnet)
- Suchmaschinenspinnen (der Code, der einer Suchmaschine eine Seite hinzufügt) schenkt automatisch mehr Aufmerksamkeit auf höhere Wichtigkeit (h1 hat die meisten, h2 hat weniger, h3 hat noch weniger, ...) Überschriften, um zu erkennen, worum es bei einer Seite geht.

Examples

Überschriften verwenden

Überschriften können verwendet werden, um das Thema zu beschreiben, dem sie vorangehen, und sie werden mit den Tags `<h1>` bis `<h6>` . Überschriften unterstützen alle globalen Attribute .

- `<h1>` definiert die wichtigste Überschrift.
- `<h6>` definiert die unwichtigste Überschrift.

Überschrift definieren:

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

Die richtige Struktur ist wichtig

Suchmaschinen und andere **Benutzeragenten** indizieren normalerweise den Seiteninhalt basierend auf Überschriftenelementen, z. B. zum Erstellen eines Inhaltsverzeichnisses. Daher ist es wichtig, die richtige Struktur für Überschriften zu verwenden.

Im Allgemeinen sollte ein Artikel ein `h1` Element für den Haupttitel aufweisen, gefolgt von `h2` Untertiteln. Falls erforderlich, wird eine Ebene nach unten verschoben. Wenn `h1` Elemente auf einer höheren Ebene vorhanden sind, sollten sie nicht zur Beschreibung von Inhalten niedrigerer Ebenen verwendet werden.

Beispieldokument (zusätzliche Absicht zur Veranschaulichung der Hierarchie):

```
<h1>Main title</h1>
<p>Introduction</p>

    <h2>Reasons</h2>

        <h3>Reason 1</h3>
        <p>Paragraph</p>

        <h3>Reason 2</h3>
        <p>Paragraph</p>

    <h2>In conclusion</h2>
    <p>Paragraph</p>
```

Überschriften online lesen: https://riptutorial.com/de/html/topic/226/uberschriften

Kapitel 41: Verwendung von HTML mit CSS

Einführung

CSS stellt HTML-Elementen auf der Seite Stile zur Verfügung. Beim Inline-Styling wird das style-Attribut in Tags verwendet und es wird dringend davon abgeraten. Interne Stylesheets verwenden das `<style>` -Tag und werden zum Festlegen von Regeln für gerichtete Abschnitte der Seite verwendet. Externe Stylesheets können über ein `<link>` -Tag verwendet werden, das eine externe CSS-Datei verwendet und die Regeln auf das Dokument anwendet. In diesem Thema wird die Verwendung aller drei Anbaumethoden behandelt.

Syntax

- `<link rel="stylesheet" type="text/css" href="stylesheet.css">`
- `<style></style>`

Examples

Externes Stylesheet verwenden

Verwenden Sie das `link` Attribut im `head` des Dokuments:

```
<head>
    <link rel="stylesheet" type="text/css" href="stylesheet.css">
</head>
```

Sie können auch Stylesheets verwenden, die von Websites über ein Content Delivery Network oder kurz CDN bereitgestellt werden. (zum Beispiel Bootstrap):

```
<head>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
</head>
```

Im Allgemeinen finden Sie CDN-Unterstützung für ein Framework auf seiner Website.

Internes Stylesheet

Sie können CSS-Elemente auch intern mit dem `<style>` -Tag einfügen:

```
<head>
    <style type="text/css">
        body {
           background-color: gray;
        }
```

```
    </style>
</head>
```

Mehrere interne Stylesheets können ebenfalls in ein Programm aufgenommen werden.

```
<head>
    <style type="text/css">
        body {
           background-color: gray;
        }
    </style>

    <style type="text/css">
        p {
           background-color: blue;
        }
    </style>
</head>
```

Inline-Stil

Sie können ein bestimmtes Element mit dem `style` Attribut formatieren:

```
<span style="color: red">This text will appear in red.</span>
```

> Hinweis: Versuchen Sie, dies zu vermeiden - CSS hat zum Ziel, Inhalte von der Präsentation zu trennen.

Mehrere Stylesheets

Es ist möglich, mehrere Stylesheets zu laden:

```
<head>
    <link rel="stylesheet" type="text/css" href="general.css">
    <link rel="stylesheet" type="text/css" href="specific.css">
</head>
```

Beachten Sie, dass **spätere Dateien und Deklarationen frühere überschreiben** . Wenn `general.css` enthält:

```
body {
    background-color: red;
}
```

und `specific.css` enthält:

```
body {
    background-color: blue;
}
```

Wenn beide verwendet werden, ist der Hintergrund des Dokuments blau.

Verwendung von HTML mit CSS online lesen:
https://riptutorial.com/de/html/topic/4536/verwendung-von-html-mit-css

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit HTML	4444, Abhishek Pandey, aea2002, ahmednawazbutt, Alexander Wigmore, Alexandre N., Amanda Ahn, amflare, Amitay Stern, animuson, Anthony Pham, Boris, bwegs, Callan Heard, ChrisD, CocoaBean, Community, Dave Everitt, Dinidu, dippas, dtyler, duskwuff, Eric Dobbs, Firix, FlyingPiMonster, geek1011, George Bailey, Gerold Broser, H Mirza, H. Pauwelyn, Harish Gyanani, Hemant Kumar, hillary.fraley, Hudson Taylor, ihavemorealts, intboolstring, Isak Combrinck, Jeffrey Lin, JHS, jmmarco, joe_young, John Slegers, Jon Chan, JonasCz, JPB, kelvinelove, Krii, Kurniawantaari, Lahiru Ashan, Lambda Ninja, Léo Martin, Leonidas Menendez, Malcolm, Matt, Matt, MC93, Michael Moriarty, mnoronha, Muntasir, Nishchay, Ortomala Lokni, Persijn, Prateek, Pyloid, Ryan Hilbert, Shannon Young, sideshowbarker, stark, Stelian Matei, Sunny R Gupta, the12, tmg, unor, user3130333, Valor Naram, Willl, Wolfgang, Zaz, zygimantus, Zze
2	Absätze	Abrar Jahin, Thomas Gerot, Valor Naram
3	Anker und Hyperlinks	Al.G., animuson, Anselm Urban, Anthony Pham, ban17, DawnPaladin, Emil, FlyingPiMonster, insertusernamehere, J F, JHS, joe_young, Jojodmo, Jones Joseph, Lambda Ninja, Matas Vaitkevicius, Nathan Tuggy, Pranav, Prateek, Raystafarian, Robert Columbia, Squidward, Steyn van Esveld , Thomas Gerot, unor, Wolfgang
4	ARIE	Bhavya Singh, Paul Sweatte, Shannon Young, Travis, unor, user30796
5	Ausgabeelement	J F, Stephen Leppik, zer00ne
6	Auswahlmenü-Steuerelemente	Ali Almoullim, amflare, animuson, GentlePurpleRain, Ilyas karim, Mosh Feu, Tot Zam
7	Bemerkungen	Ani Menon, animuson, Ashwin Ramaswami, bdkopen, ChrisD , Epodax, JHS, jkdev, RamenChef, Robert Grant, Soaring Code, Squazz, Thomas Gerot, Ulrich Schwarz, Wolfgang
8	Beschriftungselement	Anthony Pham, fredden, Josiah Keller, m_callens, Roko C. Buljan
9	Bilder	Alex, Alexandre N., andreaem, animuson, Boysenb3rry,

		Caleb Kleveter, Gabriel Chi Hong Lee, Infuzed guy, ivn, Kake_Fisk, Maximillian Laumeister, Mohd Samir Khan, Mr Lister, Nhan, Shivangi Chaurasia, Stephen Leppik, Wolfgang
10	Charakter-Entitäten	animuson, MervS, stack-learner
11	Computercode kennzeichnen	4444, Naveen Gogineni, Shannon Young, SuperStormer, Tot Zam, unor
12	Datenattribute	animuson, Community, Faegy, Infuzed guy, James Donnelly, Manish, Nathan Tuggy, Nhan, Racil Hilan, rajarshig, swatchai , unor, Yasir T
13	Div Element	animuson, Araknid, Black Mamba, Chris, Content Solutions, D M, Faust, feeela, Gal Ratzkin, JHS, MySpeed, S.L. Barth, SuperStormer, Thomas Gerot
14	Doktypen	Akshay Anand, Al.G., Angelos Chalaris, Ani Menon, animuson, Chris, Content Solutions, heerfk, mnoronha, pinjasaur, Right leg, Sumner Evans, Thomas Gerot, tmg, unor
15	Einbetten	Alexandre N.
16	Eingabesteuerelemente	Abhishek Pandey, Al.G., Alohci, amflare, Amitay Stern, Angelos Chalaris, Ani Menon, animuson, bhansa, Bob, Charlie H, Christophe Strobbe, CN, Community, cone56, Daniel, DawnPaladin, Dipen Shah, Domenic, Druzion, Edvin Tenovimas, Epodax, Franck Dernoncourt, gabe3886, geeksal , H. Pauwelyn, Henrique Barcelos, Huy Nguyen, J F, John Slegers, Kashyap Jha, Lahiru Ashan, Lankymart, Magisch, Marvin, Matas Vaitkevicius, Matt, Maximillian Laumeister, Mike McCaughan, morewry, Mosh Feu, Nathan Arthur, Nil Llisterri, Nishchay, niyasc, NoobCoder, Optimiser, Ortomala Lokni, Pi Programs, Pimgd, Prateek, Prav, Praveen Kumar, Psaniko, QoP, RamenChef, Ranjit Singh, Richard Hamilton, Robert Columbia, Roko C. Buljan, SeinopSys, Sharavnan Kv, Shivangi Chaurasia, SJDS, Squazz, Stephen Leppik, Stewartside, Sunny R Gupta, sv3k, the12, think123, Thomas Gerot, Timon, tmg, Tot Zam, trungk18, Undo, vladdobra, zzzzBov
17	Formen	Ali Almoullim, Ani Menon, animuson, Aown Muhammad, Chris Rutherfurd, Cullub, Gabriel Chi Hong Lee, Greg T, j08691, Kimmax, Luca langella, Niek Brouwer, Thomas Gerot
18	Fortschrittselement	animuson, Content Solutions, Richard Hamilton
19	Fügen Sie JavaScript-Code in HTML ein	Alexandre N., andreaem, animuson, Anselm Urban, Charles, Marjorie Pickard, MervS, Roko C. Buljan, Sildoreth,

		SuperStormer
20	Globale Attribute	animuson, Becca, unor, Zange-chan
21	HTML 5-Cache	Farhad, TricksfortheWeb, Valor Naram
22	HTML-Ereignisattribute	Paresh Maghodiya
23	IFrames	Adjit, Alexandre N., animuson, ChrisD, dorukayhan, Duh-Wayne-101, Emanuel Vintilă, J F, Ojen, Wojciech Kazior
24	Imagemaps	animuson, Lambda Ninja, RamenChef
25	Inhaltssprachen	animuson, FelipeAls, Gerold Broser, Isak Combrinck, Muntasir, Shannon Young, unor
26	Klassen und IDs	Angelos Chalaris, animuson, brandaemon, Caleb Kleveter, Community, Duh-Wayne-101, Emil, Epodax, Evan, GoatsWearHats, Ingrid Stevens, jhnance, JHS, John Slegers, lexith, Luca Putzu, Michael_B, Natalie, Nhan, Richard Hamilton, Simone Carletti, Thomas Gerot, Timothy, Tyler Zika , unor, Wolfgang, xims
27	Leere Elemente	4444, ChrisD, Thomas Gerot, unor
28	Listen	animuson, BiscuitBaker, Daniel Käfer, Grace Note, H. Pauwelyn, Jon Ericson, kcpike, Marvin, Matas Vaitkevicius, platy11, Prateek, Pseudonym Patel, Richard Hamilton, Right leg, Sayakiss, Stewartside, Thomas Gerot, tmg, Tom Johnson, unor, Zack
29	Markierungszitate	Content Solutions, mnoronha, unor
30	Medienelemente	feeela, Isak Combrinck, LisaMM, Shiva, Yossi Aharon
31	Meta-Informationen	Abhishek Pandey, Akshit Soota, Alexander Wigmore, Angelos Chalaris, Ani Menon, animuson, Anselm Urban, Bálint, bdkopen, Bookeater, Boris, coliff, Domenic, geek1011, Habel Philip, Hafidz Ilham Aji Permana, Himanshu Vaghela, insertusernamehere, jhoanna, JHS, kelvinelove, m_callens, Matt S, Michael Moriarty, Mr. Alien, Nishchay, Ortomala Lokni , Peter O., Safoor Safdar, Senjuti Mahapatra, Shannon Young , Stas Christiansen, Stephen Leppik, Ted Goas, Thomas Gerot, timmyRS, tmg, unor, VatsalSura, xims
32	Navigationsleisten	Community
33	Ressourcen verknüpfen	AA2992, animuson, Anselm Urban, Aravind Suresh, Callan Heard, Chris Rutherfurd, cone56, DawnPaladin, Domenic, feeela, Henrique Barcelos, Infuzed guy, JHS, Lambda Ninja,

		Matas Vaitkevicius, Nhan, Thomas Gerot, unor, V4karian, vladdobra
34	Schnittelemente	Andrew Brooke, Anil, animuson, Hanif Formoly, nalply, Shannon Young, SuperBiasedMan, SuperStormer, Yossi Aharon
35	Segeltuch	cone56, Richard Hamilton, Roko C. Buljan, tonethar, Trevor Clarke, user4040648
36	SVG	andreas, Black Mamba, ChrisD, HerrSerker, Patrickdev, Timothy Miller, w5m
37	Tabellen	albert, Alexandre N., animuson, Cedric Zoppolo, Eduardo Molteni, Grant Palin, J F, j08691, JHS, joe_young, Lambda Ninja, Mottie, Mr Lister, Nijin22, Prateek, PrAtik Lochawala, Praveen Kumar, Sildoreth, svarog, Ted Goas, tehciolo, Thomas Landauer, zer00ne
38	Tabindex	Content Solutions, Psaniko
39	Textformatierung	animuson, Ben Rhys-Lewis, Emil, gustavohenke, J F, Matas Vaitkevicius, Peter L., Raystafarian, Stephen Leppik, Thomas Gerot, unor, Wolfgang
40	Überschriften	Ani Menon, animuson, dippas, Evan, Infuzed guy, joe_young, MervS, Nathan Arthur, Pseudonym Patel, sasha, Thomas Gerot, unor, V-Kopio
41	Verwendung von HTML mit CSS	animuson, bdkopen, Christian Ternus, Community, Euan Williams, feeela, Jones Joseph, Michael Moriarty, Thomas Gerot, thouusten