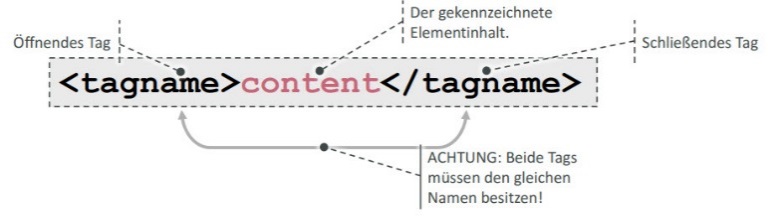
# HTML (Hypertext Markup Language)

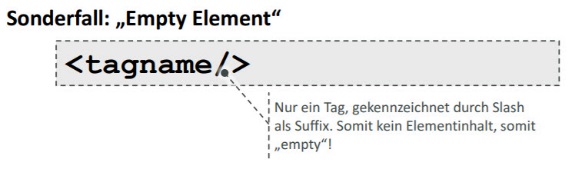
**Warum?**

Aufbau und inhaltliche Darstellung der Struktur einer Webpage

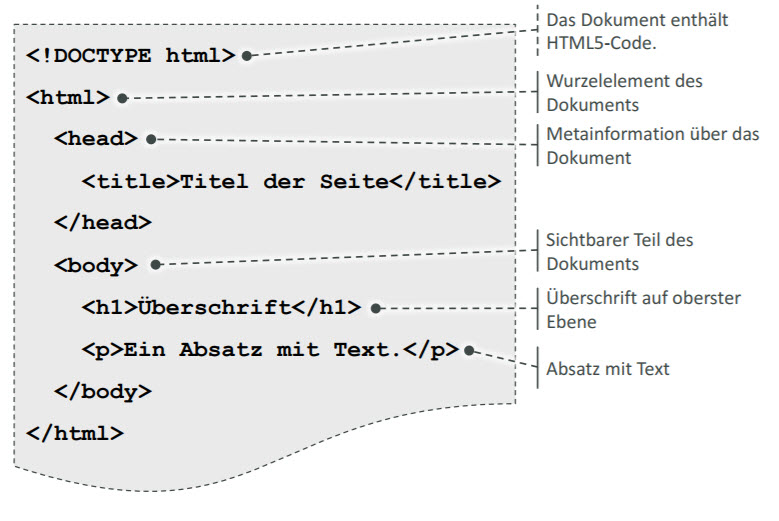
**Was?**



HTML-Elemente (Öffnende und schließende Tags) = Bausteine einer Seite



Sonderfall: Leeres Element -> Gebrauch?





🡪 Seitenstruktur

**<html>** = Root-Element, dass alle Tags als Subtags beinhaltet

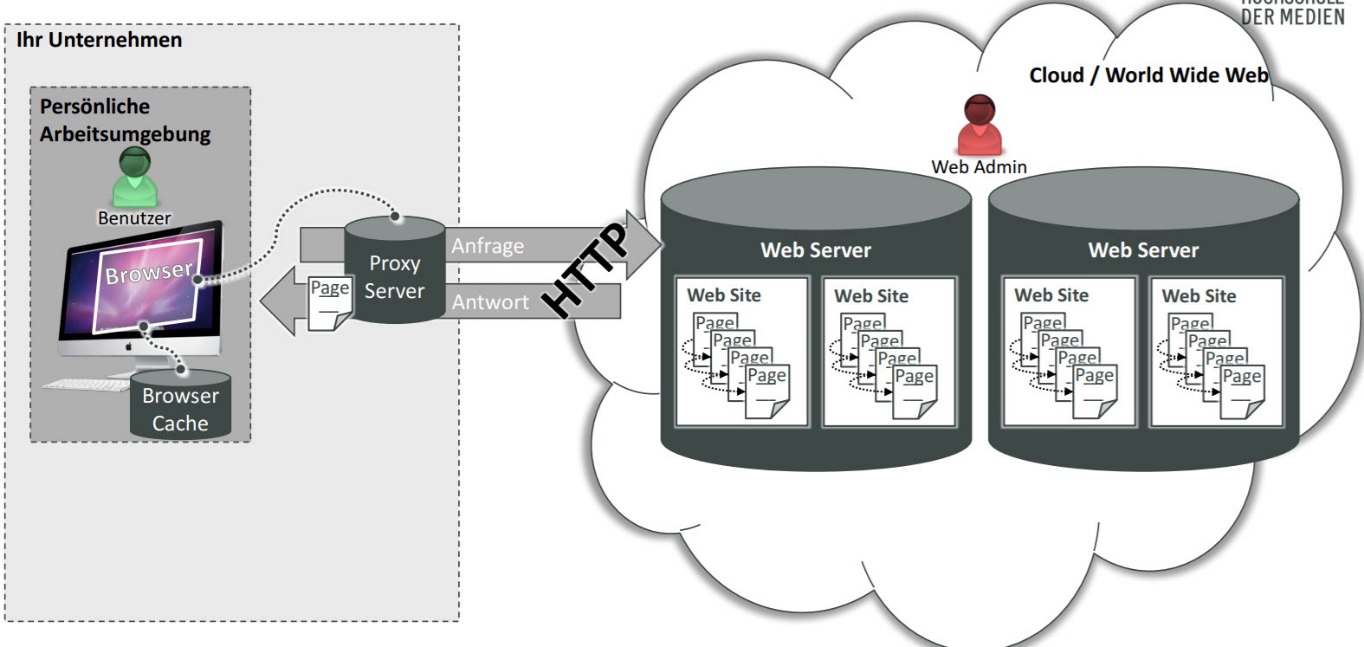
**<head>** = Metainformationen, nicht zur Anzeige gebracht

**<body>** = Enthält alle sichtbaren Elemente der Webpage als Subtags

**<h1>** = Überschriften (h1 – h6) 🡪 Je höher Zahl, desto kleiner Überschrift

**<p>** = Paragraph mit Absatz, der Text beinhaltet

**Web Server-Client Kommunikation bei Webpages**



- **Browser:**

Darstellung der HTML Tags und Interaktionsschnittstelle mit User

- **Browser Cache:**

Zwischenspeicher (lokal auf einem Rechner, ein Rechner wird bedient) für geladene Inhalte (Texte, Dateien, …) die über URL via **HTTP** abrufbar sind von Web Server für Browser

Vorteil: Schnellere Abrufbarkeit

Problem 🡪 Wenn Inhalte auf Web Server aktualisiert, werden veraltete Inhalte von Cache abgerufen

- **Proxy Server:**

Zwischenspeicher (auf Server, Rechnernetz bedienbar) für geladene Inhalte (Texte, Dateien, …) die über URL via **HTTP** abrufbar sind von Web Server für Browser auf mehreren PCs in Netzwerk

Vorteil: Schnellere Abrufbarkeit

Problem 🡪 Wenn Inhalte auf Web Server aktualisiert, werden veraltete Inhalte von Proxy Server abgerufen

- **Web Server:**

Inhalte (Styles in CSS, Bilddateien als JPG o.ä. und HTML Seite) der Webpage gespeichert und nach Request zurückgegeben

- **Web Site:**

HTML-Dokument, das von Web Server an Browser weitergeleitet wird und über URL von Browser abrufbar ist

**Anmerkung:**

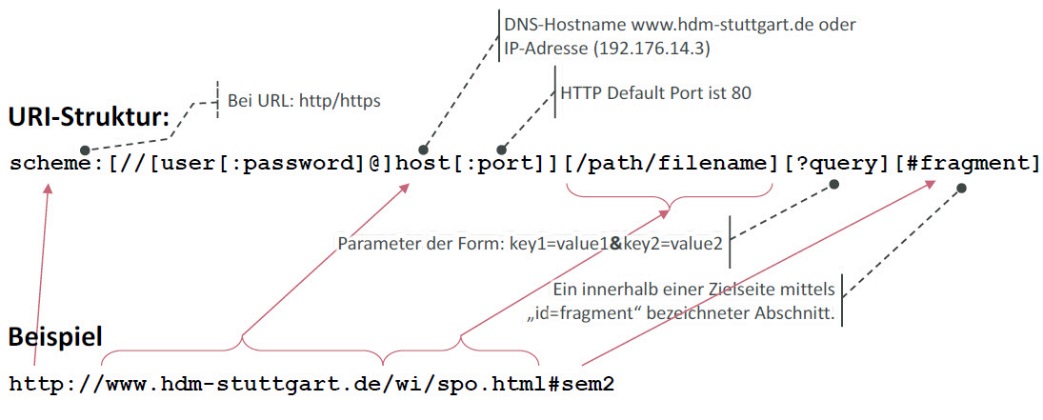
Verschiedene Websites können über mehrere Web Server hinweg durch URL Einbettung verlinkt werden!

= Hypertext!

**URL**

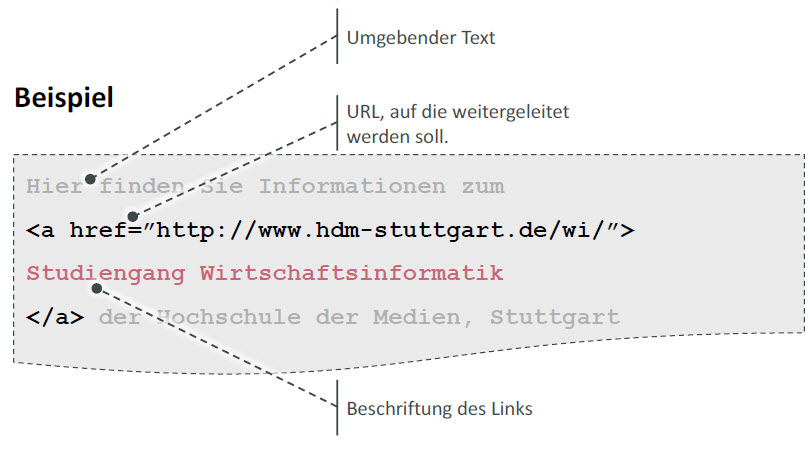
Adressierung von Websites aus Web Server vom Browser über URL Adressen

- Aufbau:



- **Hyperlinks:**

Über „href“ in **Anchor Tags** <a></a> als Attribut angeben



Absolute Pfade, oder relative zu aktuell verwendetem Server (Rootelement) angeben ?

Nur in ASCII kodierbar, somit bspw. Umlaute gesondert anzugeben

🡪 2-stelliger Hexadezimalcode mit % davor

🡪 Leerzeichen über +

🡪 Hyperlink muss nicht href und URL sein, sondern kann z.B. auch Datei sein (**<img href=““>**)

🡪 Zusätzliches **Target Attribut** gibt an, wie aufgerufener Hyperlink dargestellt werden soll

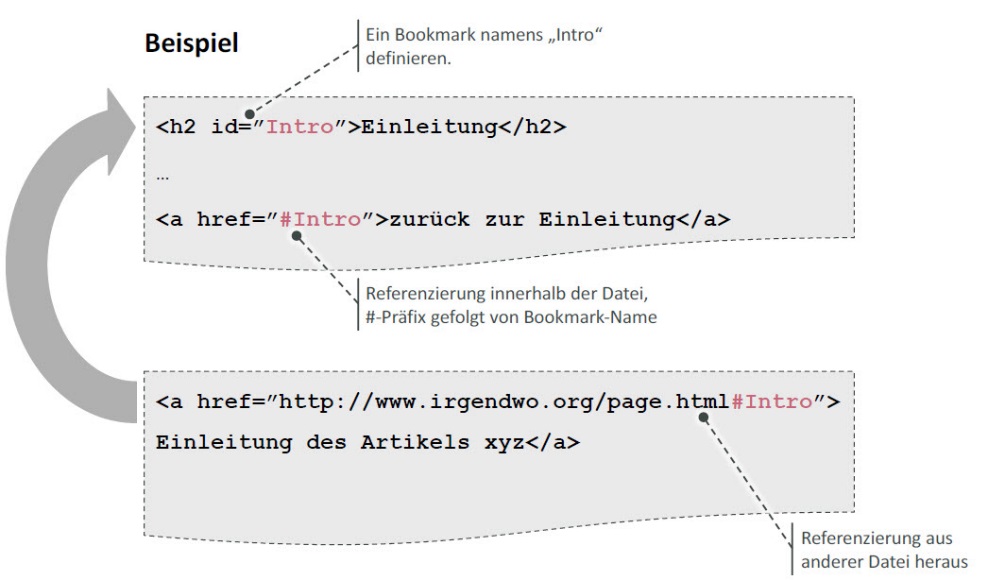
Target Attribut Werte:

* **\_blank** = Anzeige in neuem Browser Fenster oder Tab
* **\_self** = Anzeige in gleichem Fenster/Tab wie angegebener Link, entspricht gleichem Verhalten wie fehlendes Target Attribut
* **\_parent** = Anzeige in Elternframe des verlinkten Inhalts
* **\_top** = Anzeige auf oberster Ebene des aktuellen Fensters/Tabs 🡪 Ersatz aktueller Inhalte mit Inhalte des aufgerufenen Links
* **Name von Zielframe** = Fügt Inhalt des aufgerufenen Links in Frame (z.B. <div>) ein

**- Bookmarks**

Referenzieren verschiedener Stellen (auf gleicher Seiten)

Über # und Name des ID Wertes Anzugeben



**Bilder**

<img></img>

Attribute:

Src = Quelldatei

Alt = Alternativer Text, der dargestellt wird wenn Datei nicht abrufbar

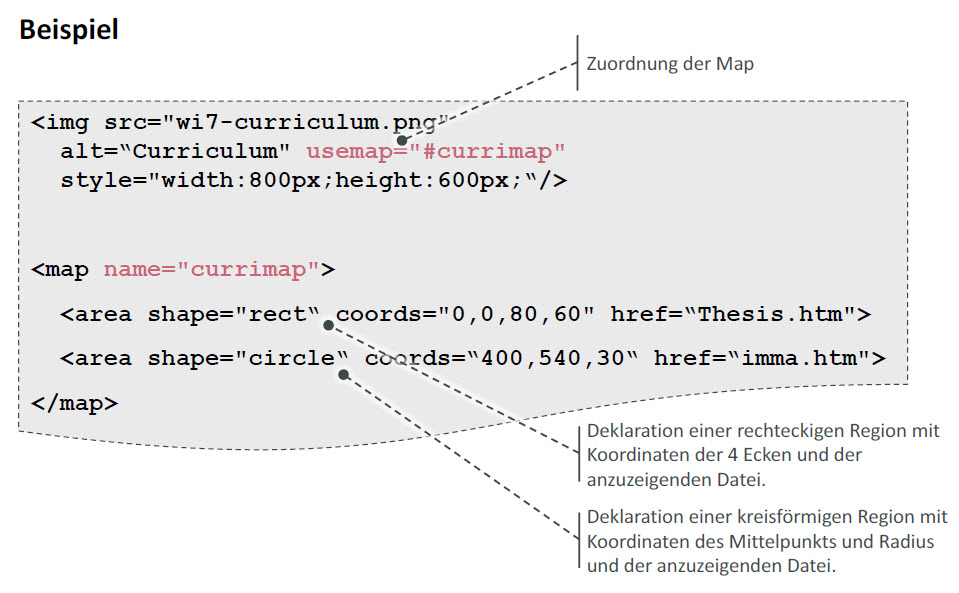
Width = Breite des Bildes in Pixeln

Height = Höhe des Bildes in Pixeln

**- Image Maps:**

Zuordnung ermöglicht Aufruf einer neuen Seite, einer weiteren Datei, o.ä., wenn Bild an einer bestimmten **Area** angeklickt wird

🡪 Zuordnung mehrerer Areas in einem Bild möglich

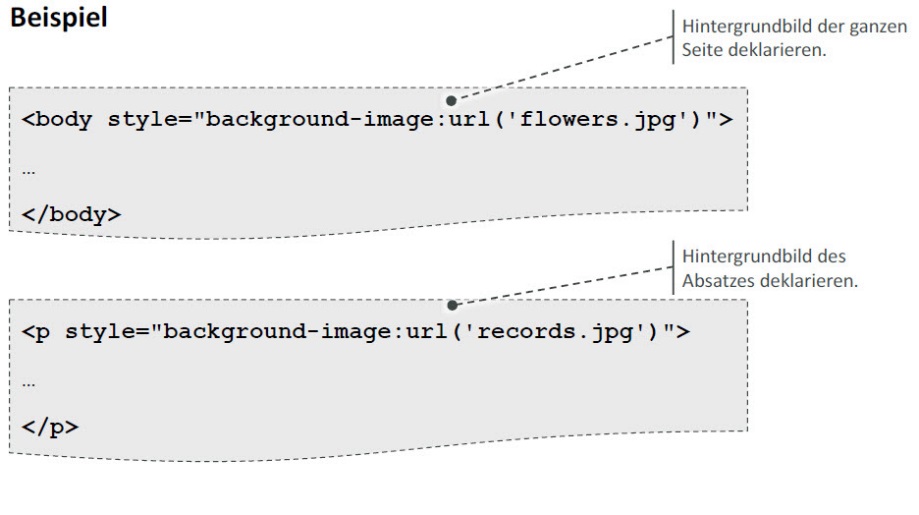


🡪 Zu Image Map „Currimap“ werden 2 Areas zugewiesen, die bei Aufruf innerhalb der angegebenen Koordinaten den über href zugewiesenen Link aufrufen

Anmerkung:

Über CSS können Bilder auch als Hintergrund von Elementen (z.B. ganze Website) verwendet werden

Aufruf über **style Attribut** in Tag des Elements, hinter das das Bild als Hintergrund gesetzt werden soll



🡪 In <body> wird Bild als Hintergrund der gesamten Website gesetzt

🡪 In <p> wird Bild als Hintergrund eines Absatzes gesetzt

**Statische vs. Dynamische Web Pages**

Statisch = Fertiggestellte, sich nicht verändernde HTML Elemente auf Web Server, die von Browser abgerufen werden

Dynamisch = Von Programm bei Bedarf erzeugt (ggf. Abruf von benötigten Informationen von Web Server), realisierbar über Java Servlets, PHP, …

Beide Web Pages über URL ansprechbar, daher keine Unterscheidung über URL Adresse möglich

**Head**

Metainformationen wie z.B.:

- Seitentitel <title> 🡪 In jedem HTML Dokument zwingend erforderlich!

Verwendung bei Aufruf von Seitentitel, wenn Seite über Suchmaschine ausgegeben oder wenn Seite als Lesezeichen in Browser hinzugefügt wird

- Style <style> 🡪 Gibt Styling Definitionen für jeweilige HTML Seite an

- Zeichenkodierung <meta> 🡪 Bspw. UTF-8

- Ergänzende Links <link> 🡪 Verlinkung zu externen Stylesheets

- Scripts <script> 🡪 JavaScript Code mit Referenz zu einer ID an der Code ausgeführt werden soll

🡪 Bei dynamischen Inhalten muss kein Server angesprochen werden!

- Base <base> 🡪 Referenziert auf URL, die als Root Element allen relativen URLs in HTML Webseite zugrunde liegt

**HTML Styling**

**- Inline CSS** = Angabe von Styling der Inhalte in jedem Tag über style Attribut

🡪 Verletzung des „**Separation of Concerns**“, da Styling nicht in separater CSS Datei abgehandelt

KLAUSUR: Vor- und Nachteile von Inline CSS

Von HTML vorgegebene Styling Elemente:

- **<b>** = Bold, steht für fetten Text

- **<strong>** = Vgl. Bold

- **<em>** = Emphasize,optisch wie italic (schräge Schrift)

- **<i>** = Italic, kursiv

- **<mark>** = Gelbe Hinterlegung

**- <small>** = Kleine Schrift

**- <del>** = Durchgestrichen, markiert Text als gelöscht

**- <ins>** = Unterstreichung, Text als eingefügt markiert

**- <sub>** = Tiefgestellter Text

**- <sup>** = Hochgestellter Text

🡪 Elemente über CSS zusätzlich modifizierbar!



**Headings**

Pro Seite wird h1 – h6 einmal verwendet

Wichtig beim Auslesen der Seite von Suchmaschinen (Auslesen der Struktur anhand von Headings)

**Paragraphs**

Abgrenzung einzelner Absätze

Erzwungene Umbrüche über <br></br>, sonst wird Text anhand von Kontext umgebrochen und ggf. an unerwünschten Stellen ausgeführt

**Absätze**

<pre></pre> (Preformatted Text)

Berücksichtigt Leerzeichen und Umbrüche in eingefügtem Text

**Horizontale Rule**

<hr> 🡪 Kein schließender Tag benötigt, da kein Inhalt!

Fügt Trennlinie ein, die thematischen Wechsel auf einer Seite darstellt

**Zitate**

**<q></q>** (Kurzzitate)

Setzt Inhalt in Anführungszeichen, also Zitation in Text

Meistens innerhalb eines Absatzes (<p></p>) gesetzt, Textfluss nicht unterbrochen, sondern nur „“ angefügt

**<blockquote>** (Längere Zitate)

Setzt Inhalt in Anführungsstriche, erzeugt neuen Absatz mit Einrückung

Kann ebenfalls URL Referenz als Quelle angeben von der zitiert wurde

**Kommentare**

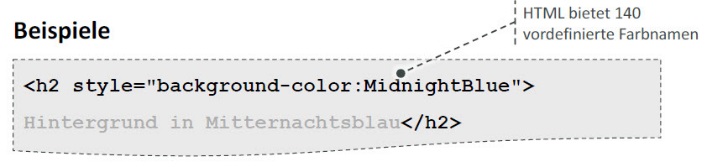
Öffnende **<!-** und schließende Klammer **-->**

Kann auch über mehrere Zeilen gehen

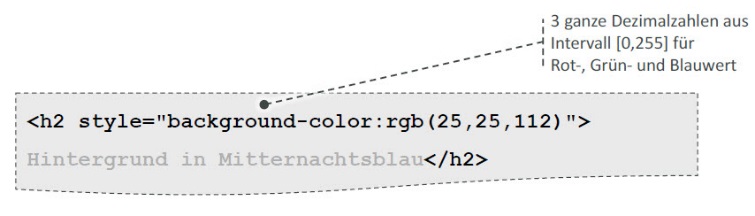
**Farbe in HTML**

Angabe in Tag durch

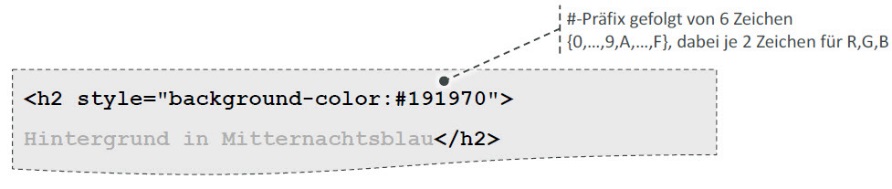
- Name der Farbe



- RGB-Wert



- Hexadezimalzahl



Oder optional über CSS Datei

**Tabellen**

Inhalte in spezieller Form anordnen

🡪 Nicht für Page Layout verwenden! (Vgl. GWT GUI Übung!)

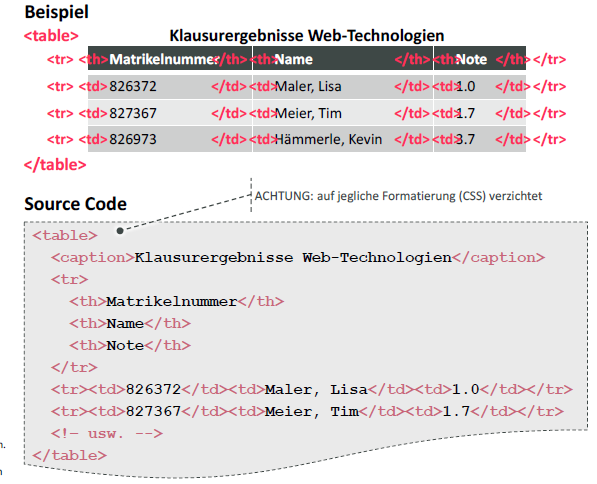
<table> = Tabelle

<tr> = Umschließt Tabellenreihe

<th> = Table Heading (Kopfzeile der Tabelle)

<td> = Table Data (Zelle in Tabelle)

<caption> = Tabellenüberschrift



**Liste**

**Unordered List:**

<ul> = Eröffnet Liste

<li> = Zeigt Aufzählungspunkt

**Ordered List:**

<ol> = Eröffnet Liste

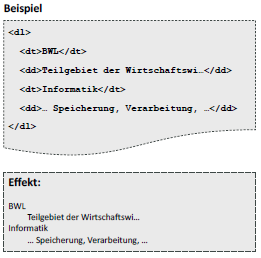
<li> = Zeigt Nummerierung

**Description List:**

<dl> = Eröffnet Liste

<dt> = Listenpunkt

<dd> = “Inhalt” der Aufzählungspunkte



**Nested List:**

Oben genannte Listenarten können ineinander verschachtelt werden



**Gruppierungselemente:**

1. Block-level Elements:

Beginnen neue Zeile & Nutzen komplette zur Verfügung stehende Breite

**<div>**

* Container für andere Elemente, zeigt Zusammenhang von Subtags auf
* Über Zuweisung von weiteren Attributen (bspw. Style) können Eigenschaften für alle Subtags gesetzt werden

**<form>**

* Formulare
* Attribut „action“ in form Tag stellt Adresse dar, zu der Eingabe geschickt wird
* Subtags häufig mit innerem Tag <input> (Usereingaben)
  + Über Attribut „type“ kann bpsw. **Text** übermittelt werden, ein **Button** zum Versenden von Informationen (Submit), **Datum, Checkboxen** (eine Auswahl)**, Radio Buttons** (mehrfache Auswahl)**, Passworteingabe, Button zum Zurücksetzen** aller Eingaben (reset), etc. angegeben werden
    - **Feldtypen** (seit HTML5):
      * „color“ zeigt Farbauswahlpalette zur Auswahl bereit
      * „number“ Dropdown-Menü an Zahlen (weiter spezifizierbar über min, max und step Attribute)
      * „range“ stellt Slider (weiter spezifizierbar über min, max und step Attribute)
      * „date“ zeigt Datepicker für User an
      * U.v.m.!
  + Attribut „name“ vergibt Input Element einen Namen
  + Attribut „Value“ stellt Text dar, der bspw. In Textbox vorausgefüllt wird oder für Buttons gesetzt wird
* Weitere Formular Elemente:
  + **Dropdown-Menü** (<select> mit <option> für Auswahlpunkte)
  + **Textarea-Element** (Vgl. Textbox, nur über mehrere Zeilen, Angabe Zeilen über „rows“ und Breite über „cols“)
  + **Button** (Optisch identisch mit Submit oder Reset Button, Unterschied: Anbindung zu JavaScript Code möglich 🡪 bspw. Erscheinendes Fenster)
* Eingabebeschränkungen: Zusätzliche Attribute (min, max, value, disabled, …) beeinflussen Verhalten von Form Elementen (Beispiel: Deaktivieren, Default Werte setzen, Zahlen zur Auswahl vorgeben)
* Datenversand über HTTP: Nutzen von GET und POST
  + **GET**: Default für HTML, in action Attribut gesetzte URL, die Formularinhalt (z.B. als Servlet) als Adresse setzt (Sichtbar in Browseradressfeld!) und nach Betätigung des Submit Buttons versendet, zusätzliches Attribut method = „get“
  + **POST**: Übermittlung vertraulicher Daten, keine Anzeige im Browser Adressfeld, keine Verwendung von URLs
* **Vgl**. Vorlesung Rathke!

**Einbetten von externen Websiten**

**<iframe>**

* Über Pixel Angabe (optional, aber hilfreich) kann Fenster in Website angegeben werden, das andere Website beinhaltet
* Eingabe von Text zwischen öffnendem und schließendem Tag wird angezeigt, wenn der abrufende Browser keine iframes unterstützt
* Benötigt Attribut src oder href

1. Inline Elements

Beginnen keine neue Zeile und nutzen minimalst möglichen Platz

Bspw. <img>, <a>

**<span>**

* Punktuelle Textformatierung über Zuweisung von Attributen in <span> innerhalb eines anderes Tags (bspw. <p>)

**Darstellung von Source Code:**

**<code>**

* Setzen von Umbrüchen mit <br> benötigt
* Sourcecode Font (Vgl. Java) mit bestimmter Breite
* Einrückung über <pre>, das <code> umschließt 🡪 Übernahme von Leerzeichen und Einrückung

**Tastatureingaben:**

**<kbd>**

* Darstellung von Shortcuts als Text ermöglicht

**<samp>**

* Konsolenausgabe von einem Programm

**Mathematische Symbole:**

Über MathML realisiert

HTML bietet <var> für einfache Formeln und Variablen (kursive Darstellung)

**HTML Entities:**

Aufruf über Entity Name (&XXXX) oder Entity Number (&#XXX)!

Bsp.

**&lt** = <

**&nbsp** (Non-breaking Space) = Leerzeichen, das in Browser nicht umgebrochen werden darf

**Diakretische Markierungen** (Erweiterung von Zeichen, z.B. mit Accents)

a&#XXX 🡪 Zu „a“ wird „à“