Weblayout

- orientiert sich am Printdesign
- Unterschied: Ausgabeformat (=Bildschirmauflösung) nicht feststehend, daher muss man entweder selbst Format definieren (z.B. 1024 x 768 Pixel) oder im Design flexibel bleiben

- Elemente einer Website:
  - Kopfbereich (head): Raum für Bannergrafiken, Navigationselemente, usw.
  - Inhalte (content): Raum für Texte, Bildergalerien, usw.
  - Fußzeile (footer): Raum für für Navigationselemente, Copyright, usw.
  - Navigationselemente: ermöglichen gruppierten Zugang zu verschiedenen Angeboten einer Website - Serviceangebote, Inhalte, Suche, usw.

- Allgemeine Regeln:
  - im Layout (Farben, Schriften, Gestaltungselemente)konsequent bleiben
  - jede Information soll mit 3 Klicks erreichbar sein
  - maximal 7 Navigationselemente je Gruppe
  - Farben: "Weniger ist mehr" (2-3 Grundfarben + hell/dunkel Abstufungen)
  - Wirkung und Bedeutung von Farben beachten

- Wahrnehmung und Gestaltpsychologie:
  - Prägnanz("Gute Gestalt"): zuerst werden einfache geometrische Formen wahrgenommen, dann Details
  - Erfahrung: Wiedererkennen vertrauter Formen
  - Negativraum ("Figur und Grund"): Kontext bestimmt Wahrnehmung (z.B. Kippbilder)
    Kontrast: Auffälliges springt ins Auge
  - Nähe: Zusammenstehendes gehört zusammen (=Gegensätze klar abgrenzen)
  - Gleichheit: Ähnliches erscheint als Gruppe (Raum-Zeit-Kontinuität)
  - Einheit und Harmonie: Eindruck visueller Verbindung schafft Gruppeneindruck
  - Geschlossenheit: Auge ergänzt Fehlendes bei angedeuteten Formen

- Regeln für Formataufteilung:
  - Goldener Schnitt (Φ)
  - arithmetische Folge (konstantes Intervall)
  - geometrische Folge (Potenzierung)

- relative Anordnung von Gestaltungselementen:
  - Position Elemente-Format: zentriert vs. optische Mitte, erhaben vs. bodenständig, ausgewogen vs. unausgewogen, symmetrisch (achssymmetrisch, punktsymmetrisch) vs. asymmetrisch
  - Position Elemente-Elemente: einfach gereiht, rhytmisch, gerastert, geometrisch, gestreut, verdichtet, gruppiert, gestaffelt

- Typografie und Webseiten:
  - Nicht mehr als 2 Schriftarten (Vorsicht vor exotischen Schriftarten!)
  - Fließtextgröße (Mengentext): 9-12 pt, Schaugrößen (Überschrift): ab 14pt; Mengentext von Überschrift deutlich unterscheidbar
  - Zeilenvorschub mind. 1,2faches der Schriftgröße
  - hoher Kontrast zwischen Schriftfarbe und Hintergrund
  - linksbündiger Flattersatz

- Typografische Gestaltungselemente
  - Absätze vs. Leerräume
  - Linien vs. Flächen

- Icons va. Hintergrundgrafiken

XHTML

- legt semantische Struktur eines HTML-Dokumentes fest
- umfasst aktuell >70 Elemente ("Tags") plus obligatorische und fakultative Attribute-Wert-Paare (abhängig von HTML-Version)
- erfordert verschachtelte Strukturen, die festlegen, welche child-Elemente in einem parent-Element enthalten sein dürfen oder müssen(Document Tree)
- leere Elemente umschließen keine Textdaten und können nicht verschachtelt werden (laden z.B. Bilder oder Skripte)

- Aufbau:
  - <elementname attribut_1="wert_1" ... attribut_n="wert_n">... </elementname>
  - Aufbau leeres Element nach XHTML: <elementname />

CSS

- Regelanweisungen für Bildschirmausgabe von HTML-Elementen
- aktueller Standard umfasst >100 Eigenschaften plus Wertangaben, jedoch sind Eigenschaften und Werte nicht beliebig mit HTML-Elementen kombinierbar
- Schema: *eigenschaft: wert;*
- erlaubt Blockkommentare nach C-Syntax
- "kaskadieren": CSS-Regeln werden je nach Wichtung auf HTML-Element angewendet und vererbt (an descendants im Document Tree)
- **Containerlayout** basiert auf **Box-Modell**

- wichtige Eigenschaftsklassen:
  - **Layout:**
    display/visibility (Anzeige und Sichtbarkeit)
    position und float (Positionieren und Elementfluss steuern)
    Box-Modell: border (Rahmen), padding und margin (Innen- und Außenabstand), width und height (Breite und Höhe)
  - **Elementformate:**
    background (Hintergrund)
    font (Schrift)
    list-style (Listen)
    Textflussformatierung
    Tabellenformatierung

- mehrere CSS-Anweisungen werden in Blöcken bzw. Regeln zusammengefasst und stehen in separaten CSS-Dateien (*.css)
- Schema Anweisungsblock bzw. Regel:

```
Selektor {
        eigenschaft_1:wert_1;
        ...;
        eigenschaft_n:wert_n
        }
```

- Selektor: weist ausgewählten Elementen bzw. Elementgruppen Anweisungsblöcke zu
- Selektortypen:
  - Universal-Selektor
  - Tag-Selektor
  - Klassen-Selektor
  - ID-Selektor
  - Kontext-Selektor bzw. kombinierter Selektor

- Implementation von CSS-Standards variiert je nach Browsertyp- und Version; insbesondere kritisch ist MS Internet Explorer einschließlich Version 7.0

Javascript

- auf Client(=Webbrowser) zugeschnittene Programmiersprache
- läuft clientseitig im Sicherheitskäfig ("Sandbox"-Verfahren)
- Objekte, Unterobjekte und deren Methoden werden mit Punktoperator verbunden: objekt.unterObjekt.methode(parameter_1,...,parameter_n) = "zugewieseneWerte"
- setzt verschiedene vordefinierte Objekte als Prototypen (Standardobjekte) ein

- Hierarchie der Standardobjekte:
  - oberstes Objekt "window" (gesamtes Browserfenster), nächstes Objekt "document" bzw. "window.document" (HTML-Dokument samt aller enthaltenen Tags) enthält Unterobjekte für bestimmte HTML-Element-Gruppe (z.B. Bilder, Formulare)

- Javaskripte stehen bei HTML-Attributen (Eventhandler, z.B. "onmouseover"), innerhalb des HTML-script-Tags (<script>...</script>) oder in separaten .js-Dateien
- *Bsp-Notation Eventhandler in HTML*
  <html-element event="javascriptfunktion(parameter_1,...,parameter_n)">

- Erstellen eigener Klassen:
  - Definition - function meineKlasse () {…}
  - Instanz - ersteInstanz = new meineKlasse()

- Implementation von Javascript variiert je nach Browsertyp- und Version

Dynamisches HTML(DHTML) und Document Object Model(DOM)

- Ziel: verändern von HTML-Elementen bzw. CSS-Anweisungen (verleiht an sich statischen HTML-Dokumenten nachträglich Interaktivität)

- DOM (Document Object Model)
  - vom W3C verabschiedete API, um bisherige propriätere DHTML-Modelle zu ersezten, um einheitlich auf Strukturelemente von Seitenbeschreibungssprachen, ihre Attribute und Textinhalte zuzugreifen
  - Elemente werden als nodes (Knoten) definiert, die baumartig in Beziehung zueinander stehen
  - Knotentypen:
    - parent (Elterknoten)
    - descendant (beliebiger Nachfahre)
    - child (unmittelbarer Nachfahre)
    - sibblings ("Geschwister": stammen vom selben Elterknoten ab)
    - Dokumentknoten (gesamte Baumstruktur)
    - Dokumentfragmentknoten (Teil der Baumstruktur)
    - Elementknoten (einzelnes Element)
    - Attributknoten (einzelnes Attribut eines Elements)
    - Textknoten (textueller Inhalt eines Elements oder Attributs #CDATA)
  - existiert in verschiedenen Versionen (DOM Level 0 – DOM Level 3)

- Bsp: Zugriff auf Wert eines Attribut eines HTML-Elementes mit Element-ID:
  - HTML: <element id="bezeichner" attribut="wert">
  - *JS-Aufruf der Methode .getElementById()*
    document.getElementbyID("bezeichner").attribut = "wert"
  - *JS-Aufruf via DOM-Level1-Syntax*

```
document.getElementById("bezeichner").getAttribute(attribut)
document.getElementById("bezeichner").setAttribute(attribut)
document.getElementById("bezeichner").firstChild.nodeValue = "wert"
```

- Bsp: Zugriff auf Wert einer CSS-Eigenschaft eines HTML-Elements mit Element-ID:
  - CSS: element#bezeichner { css-eigenschaft:wert }
  - *JS-Aufruf der Methode .getElementById()*
    document.getElementByID("bezeichner").style.cssEigenschaft = "wert"
  - *JS-Aufruf via DOM-Syntax erzeugt automatisch style-Attribut*
    document.getElementByID("bezeichner").setAttribute("attribut", wert)

Javascript-Frameworks und AJAX

- Javascript-Frameworks: Javascript-Bibliotheken, die DHTML- und AJAX-Funktionalität bereitstellen
- Schritt zu interaktiven, desktopähnlichen Webanwendungen; Grenzen zwischen Online- und Offlineanwendungen verschwinden
- Frameworks bauen teilweise aufeinander auf (z.B. beinhaltet script.aculo.us Prototype-Bibliothek, allerdings nicht die aktuellste Version; trotzdem sollte man sie nicht ersetzen, weil Probleme beim Zusammenspiel auftreten können)
- Bibliotheken wie script.aculo.us und Prototype sind frei verfügbar; große IT-Hersteller stellen oft eigene Frameworks zur Verfügung (z.B. Adobe Spry, Microsoft Ajax, Yahoo User Interface)

- AJAX (Asynchronous Javascript and XML): asynchrone Kommunikation zwischen Server und Client, bei der einzelne Struktur-Elemente, Text, Grafiken oder Nutzerdaten im Hintergrund vom Server nachgeladen werden
- AJAX ruft im Hintergrund Informationen vom Server ab, ohne dass eigentliche Dokument vollständig neu geladen wird, wie es beim klassischen Request der Fall ist

- AJAX läuft auf Basis des HTTP-Request-Objekts:
  - (FF) http_request = new XMLHttpRequest()
  - (IE) http_request = new ActiveXObject("Microsoft.XMLHTTP")

- zentrale HTTP-Request-Methoden:
  - Datenverbindung herstellen (open)
  - Zustand der Verbindung abfragen (readyState)
  - Datenformat festlegen (setRequestHeader)
  - Daten an Serverskripte senden (send)

- für die Kombination mit XML erdacht; wird jedoch auch mit anderen Datenformaten fertig, z.B. entsprechend formatierten Textdaten oder JSON
- JSON (Javascript Object Notation) kann von vielen Sprachen (z.B. PHP, C, C#, Java, usw.) kodiert werden; wird z.B. verwendet, um per PHP Daten aus einer Datenbank zu lesen, als JSON zu kodieren und dann an das aufrufende AJAX-Skript zurückgeschickt werden (Google Suggest)
- Implementation von AJAX baut auf Javascript und DOM variiert daher ebenfalls je nach Browsertyp- und Version; insbesondere kritisch MS Internet Explorer bis Version 6.0

zusätzliche Ressourcen

- **HTML und CSS**
  - **Dr.Web - Magazin für Webdesign**
    http://www.drweb.de
  - **selfhtml.org**
    Referenz zu Webtechnologien, z.B. HTML, XML, CSS, Javascript, usw.
    http://de.selfhtml.org/
  - **Verbreitung moderner Grafikbrowser**
    http://www.w3schools.com/browsers/
  - **Implementierung von CSS in modernen Browsern**
    http://css4you.de/browsercomp.html/standardbrowser/
  - **CSS Zengarden - Möglichkeiten von Webdesign mittels CSS**
    http://www.csszengarden.com/tr/deutsch/
  - **Einführung in CSS-Hacks**
    http://www.sitepoint.com/article/browser-specific-css-hacks
- **Standards und Barrierefreiheit**
  - **W3C-Referenzen für HTML, XHTML 1.0 und CSS 2.1**
    http://www.edition-w3c.de/TR/xhtml1
    http://www.edition-w3c.de/TR/1998/REC-CSS2-19980512/
    http://www.w3.org/TR/2007/CR-CSS21-20070719
  - **Valides Webdesign**
    http://validator.w3.org/
    http://jigsaw.w3.org/css-validator/
  - **Standards im Web und Acid-Tests**
    http://www.webstandards.org/
  - **Informationen über barrierefreies Webdesign**
    http://www.w3.org/TR/AERT
    http://www.macx.de/projekte/barrierefrei
  - **BIENE-Award für Barrierefreie Websites**
    http://www.biene-award.de/award/preistraeger/
- **Javascript, DHTML, DOM und AJAX**
  - **Dynamicdrive: Javascripte zum Experimentieren**
    http://www.dynamicdrive.com
  - **DHTML-Beispiele mit Erläuterungen**
    http://de.selfhtml.org/dhtml/beispiele/
  - **Javascript Frameworks: Prototype und scipt.aculo.us**
    http://www.prototypejs.org
    http://script.aculo.us/
    http://wiki.github.com/madrobby/scriptaculous/

Bongers, Frank (2007) XHTML, HTML und CSS. Galileo Press.

Erlenkötter, Helmut (2009) XML von Anfang an. 2.Aufl. Rororo.

Münz, Stefan (2009) Professionelles Webdesign. 3.Aufl. Addison-Wesley.

Wenz, Christian (2007) Javascript und Ajax. 7.Aufl. Galileo OpenPress.

Yank, Kevin / Adams, Cameron (2007) Simply Javascript. SitePoint.