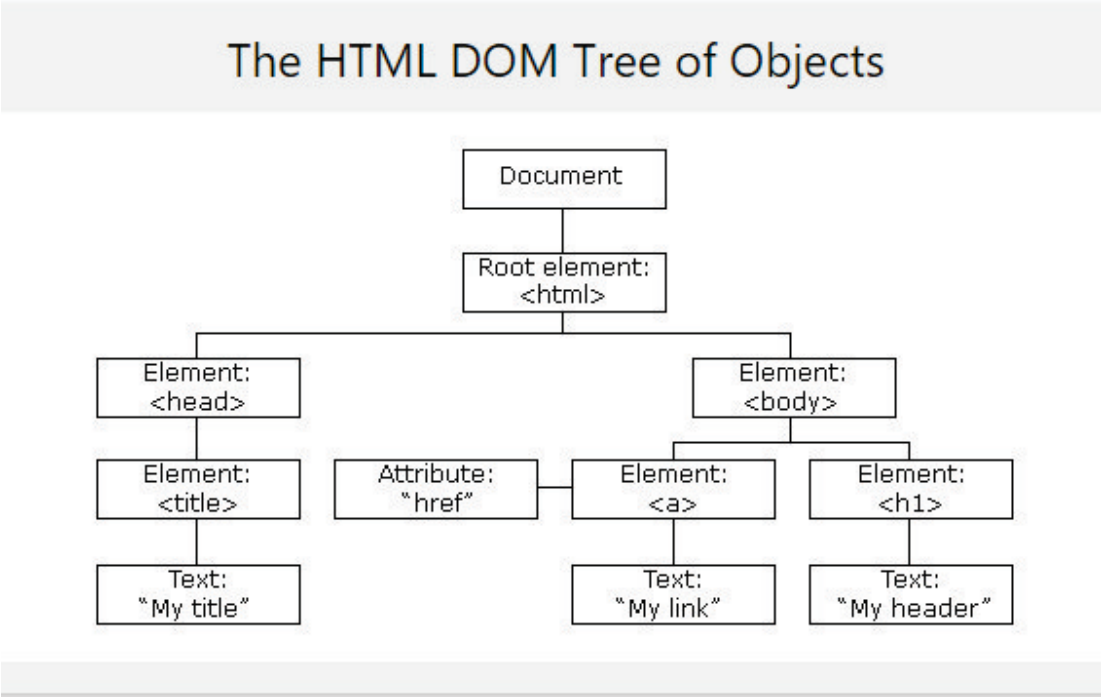
WEB Fragenkatalog

Hochschule Niederrhein WS2015

Angaben OHNE Gewähr

1) Erklären sie das W3C DOM

- Das W3C DOM steht für „World Wide Web Consortium“ und wird in drei Bereiche unterteilt.
  - ✗ Core DOM: Standardmodel für alle Dokumenttypen
  - ✗ HTML DOM: Standardmodel für HTML Dokumente
  - ✗ XML DOM: Standardmodel für XML Dokumente
- Das DOM beschreibt einen Baum mit einem Root-, Child- und den Leaf- Nodes.



- Das DOM (Document Object Module) ist plattformunabhängig und sprachneutral. Es beschreibt eine Standardschnittstelle die es Programmen und Scripten erlaubt den Zugriff auf Inhalt, Struktur und Style dynamisch zu verwalten.

2) Welche Definitionen stellt die HTML DOM Schnittstelle bereit?

- Es definiert die HTML-Elemente als Objekte
- Es definiert die Eigenschaften aller HTML-Elemente
- Es definiert Prozeduren um auf alle HTML-Elemente zuzugreifen
- Es definiert Ereignisse für alle HTML-Elemente
  - ➔ Mit anderen Worten: HTML DOM ist eine Standardschnittstelle welcher definiert wie auf den Inhalt, Struktur und den Style zugegriffen wird, Änderungen vorgenommen und Elemente hinzugefügt oder gelöscht werden.

3) Woran erkennt man ein Block- Element?

- Auf Blockelemente folgt unmittelbar ein Zeilenumbruch.

4) Was versteht man unter der Property (Eigenschaft) eine Elementes?

- Den Eigenschaften von HTML-Elementen kann ein Wert zugewiesen werden.

5) Was versteht man unter einer Prozedur über einem HTML-Element?

- Über Prozeduren erhält man Zugriff auf die DOM und somit die HTML-Elemente.
- Beispielsweise ist getElementByID eine solche Prozedur

6) Geben sie ein Beispiel für die Nutzung einer Prozedur und Property an

- document.getElementByID(``demo´´).innerHTML = „example“;
- document.getElementsByTagName(a[0]).href = „http://www.google.de“;

7) Wie lässt sich ein HTML-Element entfernen und hinzufügen?

- Hinzufügen: document.appendChild(element);
- Enrfernen: document.removeChild(element);

8) Was bezweckt man mit dem Platzieren der Script-Tags zum Dokumentende?

- Beim Aufbau einer Webpage wird jedes Element der Reihe nach verarbeitet. Platziert man die Script-Tags ungünstig, wie etwa im Header oder zu Beginn des Body, so wird das Laden des Gesamtdokumentes hinausgezögert und der User muss warten. Um das Laden transparent zu gestalten platziert man die Script-Tags zum Dokumentende. Dadurch wird die Präsentation der Webpage als erstes geladen wodurch der Eindruck entsteht, dass die komplette Seite dem User ohne Verzögerung zugänglich ist.
- Nachteile: Es kann sein dass die Script-Tags von verschiedenen Servern geladen werden oder ein Teil nicht verfügbar ist. Stellen die Script-Tags wichtige Funktionen bereit wie etwa den Warenkorb so sind diese dem User nicht zugänglich.

9) Wofür steht AJAX und wozu wird es eingesetzt?

- Asynchronous JavaScript and XML
- Stellt der User eine erneute Anfrage an Server, so werden nur die Daten aktualisiert/neu geladen, die sich seit der letzten Anfrage geändert haben.
- Dadurch wird die Netzwerklast minimiert

10) Erklären oder skizzieren sie die Prototyp- Kette

- In JavaScript wird alles und daher jede Variable, Prozedur etc. als Objekt gesehen. Alle diese Objekte erben implizit von dem obersten Objekt der Hierarchiekette, nämlich von Object. Jedes der Objekte verweist mit _proto_ auf das Papa-Objekt von dem es abgeleitet wurde und besitzt zudem eine eigene Prototyp-Eigenschaft (prototype). Prototype ist eine Art Liste der Prozeduren zugewiesen werden können. Alle Instanzen eines Objektes haben dadurch Zugriff auf die jeweiligen Prozeduren. Ist eine Prozedur nicht in der Prototypliste des Child- Objektes, so wird in der Prototyliste des nächst höheren Parentobjektes nachgeschaut. Wird die Prozedur in keiner der Prototyplisten gefunden, so gilt sie als undefiniert.

11) Geben sie ein Beispiel für einen Namensraum in JavaScript an

- var raum = new Object(); raum.attribute = „yay“;

12) Gibt es Klassen in JavaScript und wenn ja wie werden sie realisiert?

- Ja, gibt es. Allerdings nicht wie man sie aus C++, Java, C# etc. her kennt. Man spricht auch von sogenannten Pseudo-Klassen. Jede Prozedur in JavaScript kann als Klasse angesehen werden. Insbesondere kommt dabei der Namensraum zum Einsatz. In JavaScript wird alles als Objekt betrachtet, weshalb allem und jenem ein Objekt (sozusagen als Child) angehangen werden kann. Klassen machen sich dieses Feature zu nutzen indem sie ihre Attribute und Eigenschaften durch den this-Präfix nach außen hin bekannt geben. Objekte von Klassen werden mit dem new-Operator erzeugt.

13) Welche CSS-Selektoren kennen Sie? Nennen sie mindestens vier und erklären Sie.

- Universal-Selektor: * { margin: 0; }
  Hierbei wird die Formatierung auf alle DOM-Elemente angewandt.
- Typ-Selektor: p { margin: 0; }
  Hierbei wird die Formatierung auf alle spezifizierten DOM-Elemente p angewandt.
- ID-Selektoren: #id { margin: 0; }
  Hierbei wird ein einzelnes DOM-Element über die ID angesprochen. Jede ID ist einmalig und sollte daher auch nur einem einzelnen DOM-Element zugewiesen werden. ID-Attribute können zudem als Anker für einen Link im selben Dokument genutzt werden.
- Klassen-Selektor: .Kekse { color: brown; }
  Hierbei werden alle DOM-Elemente in Bezug auf eine Klasse selektiert und entsprechend formatiert. .Kekse ist äquivalent mit *.Kekse
- Mehrfach-Klassen-Selektor: .Choco.Kekse { color: brown; }
  Hierbei werden alle DOM-Elemente in Bezug zur Klasse .Choco und .Kekse selektiert und formatiert.
- Typ-Klassen-Selektor: p.Bold { font-weight: strong; }
  Hierbei wird die Formatierung auf alle DOM-Elemente vom spezifiziertem Typ p mit der in Bezug stehenden Klasse .Bold angewandt.
- Typ-ID-Selektor: a#no-decoration: a { text-decoration: none; }
  Hierbei wird die Formatierung auf alle DOM-Elemente vom spezifizierten Typ a mit der in Bezug stehenden ID #no-decoration angewandt.
- Attribut-Selektor: a[href]
  Selektiert das DOM-Element mit dem Attribut href
- Attribut-Selektor mit Wertzuweisung: a[href="blahh"]
  Selektiert das DOM-Element mit dem Attribut href welche den Wert blahh hat. Hierzu gibt es noch weitere Variationen. Siehe http://jendryschik.de/wsdev/einfuehrung/css/selektoren

14) Wofür steht HTML?

- **Hypertext Markup Language**

15 Was verstehen sie unter der späten Typ-Bindung?

- Der Datentyp wird dynamisch zur Laufzeit ermittelt.

16) Welche Probleme können bei CSS auftreten?

- Effekte unterliegen browserspezifischen Algorithmen. So ist häufig der Fall dass Webpräsentationen bei gleicher Implementierung unterschiedlich dargestellt werden.

17) Wofür steht CSS?

- **Cascading Style Sheet**

18) Was sind CSS-Pseudoklassen? Geben sie ein Beispiel an

- CSS-Pseudoklasen erkennt man an dem Doppelpunkt (:). Pseudoklassen werden verwendet um Informationen zu erfassen die man mit herkömmlichen Selektoren nicht erreichen kann oder wenn sich die Elemente außerhalb des Dokuments befinden.
- Link-Pseudoklassen: a:link { color: blue; } oder a:visited { color: green; }
- Dynamic-Pseudoklassen: a:hover oder a:focus oder a:active
- Structure-Pseudoklassen: a:nth-child(n) { color: lime; }

19) Geben sie die richtige Reihenfolge für a:hover, a:active, a:focus, a:link und a:visited an. Was passiert, wenn die Reihenfolge nicht eingehalten wird?

- a:link -> a:visited -> a:hover -> a:focus -> a:active
- Wenn die Reihenfolge nicht eingehalten wird heben sich z.B. a:link und a:visited auf. Bei a:hover, a:focus und a:active kann die eine das andere überschreiben

20) Erklären sie die Link-, Dynamic-, Structure-, und die Negation- Pseudoklassen

- Link-Pseudoklassen beeinflussen die Linkzustände
- Dynamic- Pseudoklassen sind sozusagen „Erweiterungen“ der Link-Pseudoklassen um weitere Prozeduren welche die DOM-Elemente dynamisch bei Userinteraktion verwalten.
- Structure-Pseudoklassen beziehen sich auf die Anordnung des DOM-Baums. Mit diesen Klassen kann man gezielt auf die einzelnen Knoten zugreifen und diese verwalten.
- Negation-Pseudoklassen beziehen sich auf alle DOM-Elemente die nicht ausgewählt sind. Man selektiert also diejenigen DOM-Elemente die ausgelassen werden sollen.

21) Was sind Pseudoelemente? Geben sie zwei Beispiele an

- Pseudoelemente beziehen sich auf den Inhalt der DOM-Elemente. So kann beispielsweise ein Teilstring beeinflusst werden – was vorher in HTML gesetzt werden musste wie This is a test . soll hier für eine beliebige Formatierung stehen.
- Beispiel: t his is a test → p:first-letter { color: red; } =: T(in rot)is a test

22) Mit welchen Pseudoelementen kann man Inhalt vor und nach dem eigentlichen Content erzeugen? Geben sie ein Beispiel an.

- a::before erzeugt vor dem a-Inhalt einen Inhalt.
- a::after erzeugt nach dem a-Inhalt einen Inhalt.

```
p::before {
  content:"Dieser Text wird mit dem Text von p konkatoniert ";
}
```

23) Erklären Sie REST

- REST steht für Representation State Transfer und ist eine Abstraktion der Struktur und des Verhaltens von World Wide Web. Der Vorteil von REST liegt darin, dass im WWW bereits ein Großteil der für REST nötigen Infrastruktur (z.B. Web- und Application-Server, HTTP-fähige Clients, HTML- und XML-Parser, Sicherheitsmechanismen) vorhanden ist, und viele Web-Dienste per se REST-konform sind.
- REST ist zustandslos: Jede REST-Nachricht enthält alle Informationen, die für den Server bzw. Client notwendig sind, um die Nachricht zu verstehen. Weder der Server noch die Anwendung soll Zustandsinformationen zwischen zwei Nachrichten speichern.
- Die unter einer Adresse zugänglichen Dienste können unterschiedliche Darstellungsformen (Repräsentationen) haben. Ein REST-konformer Server kann je nachdem, was die Anwendung anfordert, verschiedene Repräsentationen einer Ressource ausliefern, z.B. in verschiedenen Sprachen oder Formaten (HTML, JSON oder XML) oder auch die Beschreibung oder Dokumentation des Dienstes. Die Veränderung einer Ressource (also deren aktuellen Status) soll nur über eine Repräsentation erfolgen.
- Einheitliche Schnittstelle: REST-Nachrichten sollen selbstbeschreibend sein. Dazu zählt die Verwendung von Standardmethoden. Über diese Standardmethoden lassen sich Ressourcen manipulieren. Als Beispiel seien an dieser Stelle die HTTP-Verben genannt (GET, PUT etc.)

24) Erklären Sie JSON

- JSON steht für JavaScript Object Notation. JSON ist ein kompaktes Datenformat und ist standardmäßig UTF-8 kodiert und wurde zwecks Datenübertragung in verteilten Systemen entwickelt. Jede JSON-Definition lässt sich mit eval() in ein JavaScript-Objekt umwandeln.

25) Wie kann man in JavaScript einen Button wie in HTML <input type=“button“ /> nachbilden? Geben sie ein Beispiel an.

```
…
<form id=“myForm“ action=”myAction”>
        <input type=“text“ name=“test“ />
</form>
…
<div id=”jsButton”> </div>
…
document.getElementByID(“jsButton”).addEventListener(“click”, function() {
        document.getElementByID(“myForm”).submit();
});
…
```

26) Geben Sie ein Beispiel zur Vererbung in JavaScript an

...
function Bear() { … }

Bear.prototype = function growl() { … }

Grizzly.prototype = new Object.create(Bear()); //Here Grizzly inherits everything from Bear

function Grizzly() { Bear.call(this); } //Bear please call your own constructor

var grizzly = new Grizzly();

grizzly.growl();
…

27) Erklären sie die naive URI-Mapping im Vergleich zu REST

- Naiv: Prozeduren und deren Query-Liste (Parameter) werden an einen Link gemappt. Hierzu müssen die Prozeduren nach Außen bekannt gegeben werden. Beim Austausch von Daten sind die Value-Key-Paare in der URI sichtbar.
- REST: REST bedient sich der HTTP- Standardmethoden um Daten auszutauschen. Der Datenaustausch erfolgt nicht mehr im HTTP-Overhead sondern in dessen Body. Das hat den Vorteil dass die Query-Liste nicht mehr bei der Adressierung einzusehen ist. Des Weiteren werden die Prozeduren gegen die HTTP-Standardmethoden gemappt, was erfordert, dass die Klassenschnittstelle (Prozeduren) dementsprechend bezeichnet und nach außen bekannt gegeben werden müssen.

28) Erklären Sie Method-Dispatching

- Method-Dispatching ist eine serverseitige Einstellung der Cherrypy-API. Method-Dispatching ist ein Algorithmus welcher entscheidet welche HTTP-Standardmethode zum Austausch von Zuständen herangezogen wird. Dieser Algorithmus ist bei der Implementierung anderer REST-konformen Technologien stets anders.

29) Was ist das besondere beim Datenaustausch mit REST?

- Der Austausch von Daten mit REST ist objektorientiert. Dies bedeutet dass die Übertragung dieser Objekte client- und serverseitig zu Zustandsänderungen führt. Die Objekte beinhalten demnach alle Informationen die zur Verarbeitung der Anfrage erforderlich sind. Die Zustandsänderungen (Objekte) werden nicht zwischengespeichert, weshalb REST als zustandslos bezeichnet wird.

30) Wozu dient die Server-Push-Technologie?

- Der Server kann jetzt Daten die ein Client zur Darstellung einer Webseite benötigt ohne das weitere Clientanfragen erfolgen direkt dem Browser zuschicken (Praktisch bei Inhaltsänderungen) → Netzwerkverkehr wird entlasstet da weniger Request gespamme.

31) Warum wird REST als zustandslos bezeichnet und was bedeutet das?

- REST ist ein Teil von HTTP und HTTP ist ein zustandsloses Protokoll
- Zustandlos bedeutet, dass nach jeder Übertragung die Verbindung unterbrochen wird. Ist die Verbindung gekappt bzw. der Datentransfer beendet, wissen der Server und Client nichts mehr voneinander. Bei jedem Datentransfer muss demnach stets eine neue Verbindung zwischen Client und Server hergestellt werden. Ein wichtiges Merkmal ist außerdem, dass weder der Server noch der Client den „Zwischenstand der Dinge“ speichert.

32) Worin unterscheiden sich HTTP1, HTTP1.1 und HTTP2?

- HTTP1: Host-Header werden nicht zwingend vorausgesetzt. Nur ein Request/Response pro Verbindung. Caching musste explizit freigeschaltet bzw. angegeben werden
- HTTP1.1: Host-Header (mind. 1) werden zwingend benötigt. Es können mehrere Request/Response-Paar pro Verbindung bearbeitet werden (Pseudo-Virtuelle-Dauerhafte Verbindung). Caching wird automatisch verwaltet.
- HTTP2: Neben allem was HTTP1.1 kann bietet HTTP2 darüber hinaus eine bessere Latency indem es Kompression, eine Request/Response Pipeline und Server Push-Technologie ermöglicht.

33) Worin unterscheiden sich CSS1, CSS2.1 und CSS3?

- Zunächst will ich darauf aufmerksam machen, dass CSS hierarchisch auf seinen Vorgängern aufbaut. Dies bedeutet dass jede Version von CSS Teile der anderen enthält was eine klare Unterscheidung der Versionierung nur durch die Aufzählung der zugehörigen Features macht. Dennoch will ich sehr naiv versuchen die allergrundlegendsten Unterschiede zu benennen. Eine Tabelle der Versionierung soll anschließend mehr Klarheit schaffen. Hierzu siehe: https://de.wikipedia.org/wiki/Cascading_Style_Sheets
- CSS1: Bietet nur die grundlegendsten Formatierungsarten wie Text- und Farbformatierung.
- CSS2.1: Hinzu kam die Positionierung von Elementen, weitere Formatierungsarten wie Schatten und zahlreiche Selektoren
- CSS3: Anders wie die Vorgänger wurde CSS3 modularisiert. Die Modularisierung erlaubt es Teile des CSS effizienter zu pflegen und fortwährend auszubauen. Die wohl größten Änderung/Neuerungen waren die Erweiterung von Selektoren, Das Hinzufügen von Transformationen und Animationen sowie zahlreiche weitere Style-Formatierungsarten.

34) Was ist der Unterschied zwischen http://www.domain.de und http://domain.de?

- Es gibt keinen Unterschied.
- Bei www handelt es sich um eine Sub-Domain. Damals und auch heute noch gängig werden neben der Webpage auch andere Service über HTTP angeboten wie etwa http://ftp.domain.de, http://smtp.domain.de, http://mail.domain.de, dem allseits bekannten http://www.domain.de etc. Heutzutage werden bzw. können die Sub-Domains weggelassen werden da die Dienstzuordnung über mächtigere Protokolle (TCP/UDP) verwalten wird.

35) Was verstehen sie unter der späten Typ-Bindung?

- Bei der späten Typ-Bindung wird der Datentyp dynamisch zur Laufzeit ermittelt. Dies hat den Vorteil dass ein Quellcode nicht erst compiliert werden muss sondern Interpretiert wird. Der Nachteil ist allerdings der dass Applikationen die interpretiert werden deutlich langsamer sind wie ihre kompilierten Artgenossen. Cherrypy bietet hiezu eine hybride Lösung an, indem der interpretierte Code als Bytecode gespeichert wird. Wird die Applikation erneut ausgeführt so greift Cherrypy auf den gespeicherten Bytecode zu und übersetzt ihn in die Maschinensprache. Hierbei entfällt eine erneute Interpretation der Quellcodes.

36) Geben Sie ein Code-Beispiel für URI-Mapping an

```
1   import cherrypy
2
3   class URI_Mapping(object):
4
5       @cherrypy.expose
6       def index(self):
7           return "Ausgabe von Hauptseite via URI Mapping";
8
9   if __name__ == '__main__':
10      cherrypy.quickstart(URI_Mapping());
11
```

37) Geben sie ein Code-Beispiel für Method- Dispatching (REST) an

```
1   import cherrypy
2
3   class Method_Dispatching(object):
4
5       exposed = True
6
7       def GET(self):
8           return "Method Dispatching";
9
10
11  if __name__ == '__main__':
12
13      conf = {
14
15          '/': {
16                'request.dispatch': cherrypy.dispatch.MethodDispatcher(),
17                'tools.response_headers.on': True,
18                'tools.response_headers.headers': [('Content-Type', 'text/plain')],
19            }
20      }
21
22      cherrypy.quickstart(Method_Dispatching(), "/", conf);
23
```

38) Geben sie ein Code-Beispiel für gemischtes URI-Mapping und Method-Dispatching an

```python
import cherrypy

class URI_Mapping(object):

    @cherrypy.expose
    def index(self):
        return "Ausgabe von Hauptseite via URI_Mapping";

    @cherrypy.expose
    def link(self):
        return "Ausgabe von link via URI_Mapping";


class Method_Dispatching(object):

    exposed = True

    def GET(self):
        return "yay";


if __name__ == '__main__':

    conf = {

       '/link': {},

       '/m_disp': {
            'request.dispatch': cherrypy.dispatch.MethodDispatcher(),
            'tools.response_headers.on': True,
            'tools.response_headers.headers': [('Content-Type', 'text/plain')],
        }
    }

    mixed = URI_Mapping()
    mixed.m_disp = Method_Dispatching()
    cherrypy.quickstart(mixed, "/", conf);

```

39) Erklären sie Event- Bubbling und Capturing

- Bubbling: Man stelle sich eine Matroschka vor. Es wird zunächst das innerste Event getriggert und anschließend das nächst höhere Event in der Rangordnung bis hin zum äußersten Eventhandler.
- Capturing: Wie Bubbling nur umgekehrt.

40) Wodurch unterscheiden sich die XHTML-Elemente div und p?

- Beides sind Block-Elemente mit dem Unterschied, dass das div Block-Elemente beinhalten kann und p hingegen nicht.

41) Was versteht man unter einer XML Processing Instruction?

- XML Processing Instruction haben eine große Ähnlichkeit zu den HTML- Tags, sind aber selber keine DOM-Elemente wie man sie von HTML her kennt. Eine Deklaration gleicht der Anweisung nahezu identisch.
  Hier ein Deklarationsbeispiel: <?xml version="1.0" encoding="UTF-8" ?>
  Solche XML Anweisungen richten sich in der Regel an den Parser und beinhalten Informationen zur Verarbeitung des Dokumentes.

42) Welche Wirkung hat der CSS-Stil float, wovon hängt er ab und welche Einstellungen gibt es?

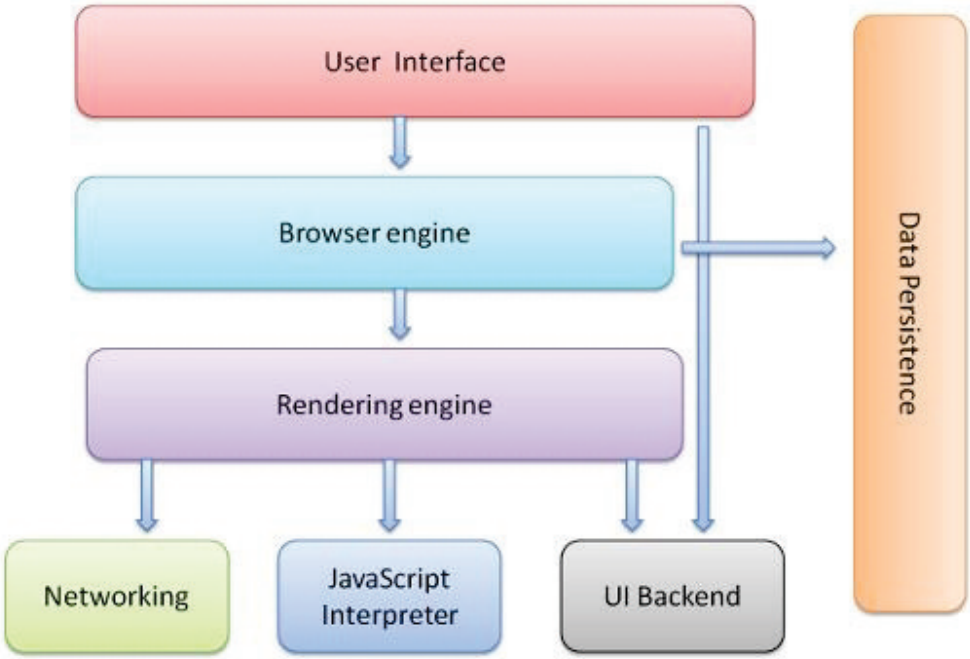
- Die float-Einstellung ist abhängig bzw. relativ zu seinem Parent. Dabei wird das gefloatete Element aus seinem normalen Elementfluss herausgelöst. Das bedeutet, dass ein float keinerlei Einfluss auf sein Elternelement oder nachfolgende Blockelemente besitzt. Im Einzelnen hat dies folgende Auswirkungen: Das Elternelement eines floats wird diesen nicht umschließen. Floats können also aus ihrem Elternelement nach unten herausragen. Es kann auch vorkommen, dass das Elternelement zu verschwinden scheint, wenn es nur Floats als Nachfahren besitzt. In diesem Fall gibt es keine Elemente mehr, die innerhalb des Elternelements Platz einnehmen könnten. Blockelemente, die neben einem Float dargestellt werden, nehmen ohne Berücksichtigung des Floats die volle Breite ihres Elternelements ein. Sie werden also teilweise von einem floatenden Element verdeckt. Die Besonderheit bei floatenden Elementen liegt in der Fähigkeit, Inlineelemente verschieben zu können. Dabei werden die Zeilenboxen, in denen Texte, Bilder oder andere Inlineelemente dargestellt werden, soweit verkürzt, dass diese um die Float-Box herumfließen können. Elementen den das Attribut float zugewiesen wird werden zu Blockelementen. Anhand der Einstellung kann dabei die Positionierungsart des neuen Blockelementes festgelegt werden. Standardmäßig folgt auf ein Blockelement immer ein Zeilenumbruch. Mit float: left; beispielsweise würde der Zeilenumbruch aufgehoben und stattdessen links angeordnet werden.
- Einstellungen: float: left, right, initial, none, inherit

43) Welche speziellen Tests sind bei Web- Applikationen im Vergleich zu Desktop- Applikationen erforderlich?

- W3C Validation: Ein HTML- Validator
- HTML-Hint (htmlhint.com): Ermöglich Teile/Auszüge des Markup, CSS und JS zu debuggen
- Die Funktionalitäten von Web-Applikationen müssen in Anhänglichkeit der Browserwahl geprüft werden. Hintergrund ist der dass die Algorithmen der Browser die zur Darstellung, Verarbeitung und Bearbeitung der Informationen unterschiedlich implementiert sind, wodurch sich die Web-Präsentation von Browser zu Browser unterscheidet.

44) Was ist ein Webbrowser?

- Ein Browser ist eine Applikation die dem Benutzer als User-Interface dient. Im Kern besteht der Browser aus mehreren Applikations-Schichten welche für die Verwaltung und Darstellung der angefragten Informationen verantwortlich sind.
  Dazu gehören:
  - Die Benutzeroberfläche
  - Die Browser-Engine
  - Die Rendering-Engine
  - Das Netzwerk
  - Der Datenspeicher
  - Der JavaScript-Interpreter und Parser



45) Sie sollen drei Bereiche, die jeweils als div-Element realisiert werden, auf einer Webseite als Kopfbereich (id="kopf") mit der Höhe 50 Pixel, Inhaltsbereich (id="inhalt") und darunter einen Fußbereich (id="fuss") mit einer Höhe von 50 Pixel anordnen. Die Bereiche sollen den sichtbaren Bereich der Webseite vollständig ausfüllen (keine Ränder und Abstände). Geben Sie die CSS- Stilregeln so an, dass der Aufbau der Webseite auch bei einer Größenänderung des sichtbaren Bereichs der Webseite erhalten bleibt.

```html
<!doctype html>

<html lang="de">
    <head>
        <title> WöBseitÖ </title>
        <meta charset="UTF-8">
        <link rel="stylesheet" type="text/css" href="css.css">
    </head>
    <body>
        <div id="header"></div>
        <div id="content"></div>
        <div id="footer"></div>
    </body>
</html>
```

```css
body > div[id="header"] {

    background-color: green;
    position: absolute;
    left: 0;
    top: 0;
    right: 0;
    height: 50px;

}
```

```css
div[id="content"] {

    background-color: orange;
    position: absolute;
    left: 0;
    top: 50px;
    right: 0;
    bottom: 50px;

}
```

```css
*#footer {

    background-color: blue;
    position: absolute;
    left: 0;
    bottom: 0;
    right: 0;
    height: 50px;

}
```

46) Geben sie eine Beispiel für die CSS-Selektoren UND, ODER und INNERHALB an

- UND: .dies.das → gültig für class="dies", class="das", class="dies das"
- ODER: .dies, .das → gültig für class="dies" oder class="das"
- INNERHALB: .dies .das → .das ist ein Child der Klasse .dies und ist ihr untergeordnet

47) Beschreiben sie die CSS-Stilregel inline, internal und external

- inline: Bezieht sich auf ein DOM-Element und wird mit der Style- Property gesetzt
   blahh
- internal: Wird im head-Bereich des Dokumentes angegeben
  <style> ..formatierung.. </style>
- external: Die CSS-Datei befindet sich außerhalb des Dokumentes und wird im head-Bereich durch einen Verweis importiert. <link rel="stylesheet" href="style.css">

48) Was bewirkt @import „stylesheet.css"?

- Mit @import kann man in einem CSS-Stylesheet ein anderes CSS-Stylesheet importieren.

49) Auf welche zwei Arten kann man eine Font in CSS importieren die nicht auf dem lokalen System installiert ist?

- @import url(_link_zum _font_)
  h1 { font-family: fontname; }
- @font-face {
  font-family: fontname;
  url(_link_zum _font_) format(„truetype");
  }

  h1 { font-family: fontname; }

50) Was bewirken @media, @document und @supports? Geben sie je ein Beispiel an

- @media: Richtet sich an die genutzten Medien wie dem Monitor oder dem Drucker und gibt die Sytelformatierung vor.
  @media screen { p { font-size: 1.5em; } } → p wird mit 1.5 bei der Bildschirmausgabe gewichtet.
  @media print { p { font-size: 1.5em; } } → p wird mit 1.5 bei dem Drucker gewichtet
- @ducument url(_link_zum_dokument_): Richtet sich an externen Dokumente und formatiert sie dementsprechen. Achtung wird nicht von jedem Browser unterstützt!
  @ducument url(_link_zum_dokument_) { p { font-family: arial 1.5em; } }
- @supports: Prüft ob der Browser eine bestimmte Formatierung unterstützt. Wenn ja wird diese angewandt.
  @supports (text-shadow) { h1 { text-shadow: 1px 2px 1px black; } }