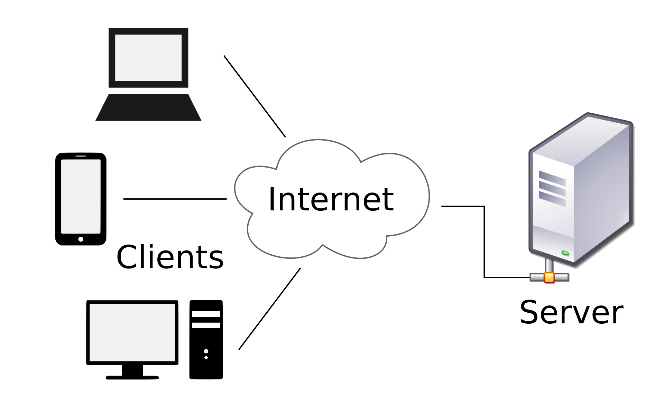
# Изпитна тема No 14: Интернет програмиране

* Мрежови протоколи (IP, TCP, UDP).
* Видове HTTP заявки (GET / POST / PUT / DELETE / PATCH).
* Клиент-сървърна комуникация. Основни тагове в HTML.
* Селектори и основни правила в CSS. Създаване на семантични страници.
* Създаване на адаптивно (responsive) оформление на страници.
* Увод в JavaScript – работа с обекти и събития.
* Принципи и манипулиране на DOM.

Дидактически материали: Компютър с подходяща версия на интегрирана среда за разработка от изучаваните и уеб браузър. Например, Visual Studio, Visual Studio Code, SublimeText, Atom и др.

1. Обяснява и диференцира различните протоколи.
   1. IP, Internet Protocol, е network-layer протокол който осъществява връзка между апликации, поддържа TCP и UDP. То е предназначен да описва адреси и връзки между компютри в една мрежа.
   2. TCP, Transmission Control Protocol e transport-layer протокол, първо се осъществява връзка между двата края, и се пращат пакети последователно. TCP връзки са едно към едно. Предназначен е за употреба в апликации, нуждаещ се от надеждна връзка, за сметка на скорост, загубени пакети се пре-пращат.
   3. UPD е transport-layer протокол, пакети те се пращат към дестинацията без да се осъществи предварителна връзка. UDP връзките могат да бъдат едно към едно или чрез Broadcasting, едно към много. Предназначен е да предостави информация по най-бързия начин, независими как, от къде или дали дестинацията е готов да приеме данни, UDP връзките не са много надеждни и често губят пакети, които по стандартна имплементация, не се пращат на ново.
2. Дефинира понятието HTTP заявка, прави изводи за различните HTTP методи и избира метод за конкретна ситуация.
   1. HTTP заявка е процес за искане на информация или пращане на информация към даден Уеб Сървър. Това е основния начин на комуникация в модерния уеб.
   2. GET заявка е за искане на информация. Пример за това би било искането на информация за дадена страница в сайт.
   3. POST заявка е за изпращане на нова информация. Това би било пращането на съобщения или коментар в блог или чат апликация.
   4. PUT заявка е за пращане на бинарни ресурси, като файлове. Това би било качването на снимка в социална медия.
   5. DELETE заявка е за изтриване на даден ресурс. Пример за това би било изтриването на коментар от социална медия.
   6. PATCH заявка е за промяна на даден ресурс. Това би било промяната на текста на даден блог пост.
3. Обяснява и представя графично клиент-сървърната комуникация.
   1. 
   2. Клиенти се свързват с централен сървър чрез локална или интернет връзка, и комуникират директно и само със сървъра.
   3. Комуникация към други услуги или клиенти се осъществява чрез посредничество на сървъра.

1. Различава смисъла на употребата и необходимостта от HTML, CSS и JavaScript.
   1. HTML представлява структурата на една страница, без нея няма как да се структурира информацията. Това би било скелета и мускулите на един човек.

* Стандартното разширение на HTML страница е .html и .htm. Разширението .htm се използва като алтернатива на .html в някои операционни системи и сървъри, които не приемат четирибуквени разширения.
* Разширенията .shtml и .shtm се използват за HTML страници съчетани с SSI (Server Side Includes).
* Разширението .xhtml се използва за страници написани на XHTML (Extensible HyperText Markup Language).

HTML (съкращение от термина на английски: HyperText Markup Language, в превод „език за маркиране на хипертекст“) е основният маркиращ език за описание и дизайн на уеб страници. HTML е стандарт в Интернет, а правилата се определят от международния консорциум W3C. Текущата версия на стандарта е HTML 5.0.

Елементи на езика HTML - Под елемент се разбира команда на HTML. Елементът дава възможност да се промени начина, по който изглежда даден текстов документ, когато той се интерпретира от WWW клиент (Web Browser). По този начин всеки един HTML документ се изгражда от HTML елементи. Някои от елементите са част от структурата на HTML документа, останалите представят съдържанието на уеб страницата. HTML елементите се изписват с отварящ таг, затварящ таг и съдържание помежду им.

<отварящ таг> съдържание </затварящ таг>

* 1. CSS описва визуалната му презентация, без нея всичко би било чист текст. Това би било кожата и косата на един човек.

CSS (Cascading Style Sheets) е език за описание на стилове – използва се основно за описване на представянето на документ, написан на език за маркиране. Най-често се използва заедно с HTML, но може да се приложи върху произволен XML документ. Официално спецификацията на CSS се поддържа от W3C.

Използвайки CSS, настройките за форматиране могат да бъдат поставени в един-единствен файл и тогава промяната ще бъде отразена едновременно на всички страници, които използват този CSS файл. CSS позволява да се определя как да изглеждат елементите на една HTML страница – шрифтове, размери, цветове, фонове, и др. CSS кодът се състои от последователност от стилови правила, всяко от които представлява селектор, последван от свойства и стойности. Селекторите се използват за да покажат към кои елементи на HTML документа трябва да бъде прилаган съответният стил. Съществуват много видове селектори. Някои селектори позволяват постигане на динамичност на страницата до определена степен. Например само с помощта на CSS могат да бъдат направени изскачащи менюта, хипервръзки, които при посочване променят цвета си и др.

* 1. JavaScript добавя интерактивност и интелигентност на страницата, без нея няма как да има интерактивност. Това би било мозъка и нервната система на един човек.

JavaScript е интерпретируем език за програмиране, разпространяван с повечето уеб браузъри. JavaScript е програмен език, който позволява динамична промяна на поведението на браузъра в рамките на дадена HTML страницата. JavaScript се зарежда, интерпретира и изпълнява от уеб браузъра. JavaScript функции могат да се свържат със събития на страницата (например: движение/натискане на мишката, клавиатурата или елемент от страницата, и други потребителски действия). Прието е JavaScript програмите да се наричат скриптове.

1. Дефинира и използва коректно HTML тагове.
   1. Прекалено много са, тук да се ползва онлайн ресурс.

<!DOCTYPE> Този таг стои на първо място в html документа. Дори преди <html> тага. <!DOCTYPE> не е декларация на html таг, той е инструкция към уеб браузъра, показваща на коя версия на HTML е написана съответната html страница.

<html></html> Този таг създава html документ. Той е коренът на html документа и в него се съдържат всички други елементи.

<meta> Мета информацията е тази, която се отнася за данните. Този таг съдържа съдържание за документа. Информацията, съхранявана в този таг, не се показва от браузъра. Мета елементите обикновено специфицират описание на страниците, ключови думи, автор на документа, модификация и други. Тази информация се използва от браузъра за това как той да показва съдържанието или как да презарежда страницата.

<head></head> Между отварящия таг <head> и затварящия таг <head> стоят всички елементи като заглавие на документа <title>, скриптове <script>, стилове <style>, мета информация <meta>, линкове <link> и други.

<body></body> Този таг дефинира тялото на документа и съдържа цялото му съдържание като текст, хиперлинкове, изображения, таблици, списъци, форми и други.

<h1></h1>,…,<h6></h6> Заглавията в HTML се дефинират с тага <h1> и подзаглавия до <h6>. Търсачките използват заглавията, за да индексират структурата и съдържанието на вашите уеб страници. Потребителите скриват страниците ви по заглавията си. Важно е да използвате заглавия, за да покажете структурата на документа.

<p></p> Този таг дефинира параграф. Браузърът автоматично добавя отстъпи и разстояние, когато стигне до такъв таг. Разстоянията могат да се променят чрез CSS. HTML Линкове/Връзки - Линковете в HTML се дефинират с тага <a>

<img> Тагът дефинира изображения в страницата. <img> тагът има следните атрибути: src и alt. Изображенията не се поставят в html страницата, те се показват, като се укаже линк към тях. Ако искате да свържете изображение към друг документ, просто сложете <img> тага в <а> тага.

<а></а> В този таг се поставят хиперлинкове, които обикновено се използват за връзка на една html страница с друга. Най-важният атрибут на <a> тага е href, след който се изписва линкът. Линковете ще се показват от всички браузъри, като те могат да бъдат посетени, непосетени и активни.

HTML Line Breaks - Елементът HTML <br> дефинира прекъсване на реда. Използвайте <br> ако искате прекъсване на линия (нов ред), без да започнете нов параграф. Тагът <br> е от типа така наречени празни тагове, което означава, че няма краен/затварящ маркер.

Повечето браузъри ще покажат правилно HTML дори ако се забрави крайният маркер.

1. Задава свойства на HTML компонентите чрез CSS.

Каскадните стилове представляват съвкупност от правила, прилагани за оформяне на определени елементи на страниците на сайта. Технологията на каскадните стилове (CSS) позволява съхраняване на цялата информация за разпределението на страницата, използваните на нея шрифтове, цветове, стилове за оформяне на меню и т.н. в определено място (един или няколко файла). CSS (Cascading Style Sheet) предоставя допълнителна функционалност и възможности в изграждането на уеб страници. Той позволява на дизайнера да контролира параметрите за визуализиране на всеки един HTML таг. CSS дава възможност за дефинирането на класове и за използването на общ стил (CSS файл) от всички страници в даден сайт.

Вмъкване на CSS в HTML документ

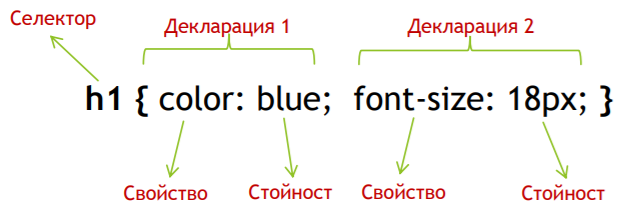
Съществуват четири метода за вмъкване на CSS в HTML документ:

1. Добавяне на CSS към HTML таг <h2 style="color:red;background:black;">This is a red heading with a black background</h2>

2. Добавяне в <head> частта на HTML документ <style media="screen" type="text/css"> Описание </style>

3. Свързване на HTML документа с отделен CSS файл <link rel="stylesheet" type="text/css" href="mystyles.css" media="screen" />

4. Вмъкване на CSS в CSS @import "newstyles.css";



* Селекторът посочва елемента, за който се отнася правилото.
* Декларацията съдържа точната стойност на свойството. Всяка декларация завършва с ; .
* Една декларация или последната могат да бъдат записани без ; .
* Свойството е характеристика на елемента. Свойство color в декларация 1 задава цвят

на текста. Свойство font-size в декларация 2 задава размер на шрифта.

* Стойността е състояние на свойството. Всяко свойство може да приема точно

определени стойности.

Селектор на елемент - Това е най-простият тип селектор и много списъци със стилове се състоят предимно от селектори на елементи.

ПРИМЕР:

body {background-color: lightcyan;}

h2 {color: darkblue;}

p {font-size: 120%;}

Когато различни селектори имат еднакви декларации, те могат да бъдат групирани – изброяват се със запетая. В примера заглавия h1 и h2 имат еднакво свойство – червен цвят на текста.

ПРИМЕР:

h1 {color: red;}

h2 {color: red;}

h1, h2 {color: red;}

Използване на универсален селектор - Позволява бързо селектиране на група от елементи и прилагане на стилово правило. За целта използва символа \*.

ПРИМЕР: \* {color: purple;}

Каскадност - Каскадността на стилове е едно от съществените свойства в CSS. То се основава на следнте правила:

1. Показване на стил - Само едни стил може да бъде приложен върху даден елемент.

2. Място в списъка със стилове - При множество стилове приложени върху даден елемент, се изпълнява последния в списъка.

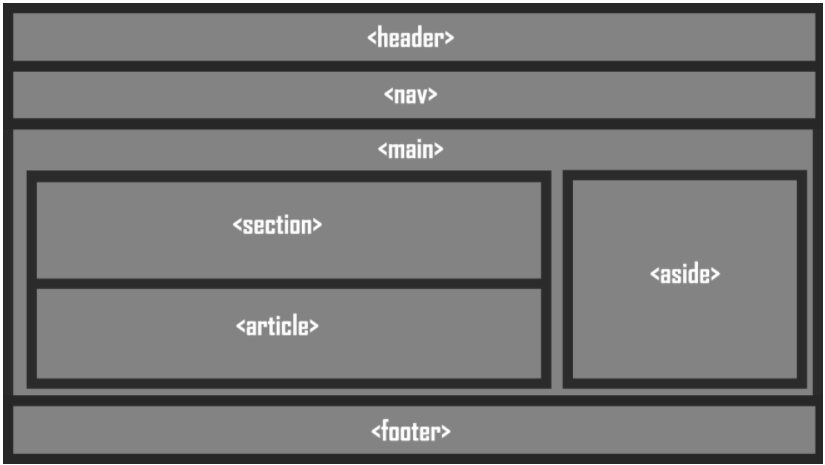
3. Използване на ключвова дума !important - Ключова дума !important позволява даден стил да се изпълни независимо от мястото му в списъка.

Наследяване - Елементите в HTML страница са организирани в йерархична структура. Родителските елементи съдържат в себе си други елементи, наречени деца. В йерархията на HTML документа най-висшият родителски елемент е <html>. Съответно <head> и <body> са непосредствени деца на <html>. Всички елементи, които се намират в <body> са негови деца и наследяават неговите свойства. В CSS свойствата се наследяват от родителските елементи към децата.

* 1. Компонента се избира чрез един от следните начини:
     1. Името на тага
     2. Class – Класа даден на компонента при създаване или по-късно чрез JavaScript.
     3. Id - Идентификатора даден на компонента при създаване или по-късно чрез JavaScript.
     4. Релативно местоположение в документа – Описват се взаимоотношенията на компоненти чрез техните имена, класове и идентификатори.

1. Описва и демонстрира употребата на семантични елементи за създаване на семантична страница.
   1. Семантични елементи и страници са документи или частни на документа, описани подробно по начин разбираем за машини/автоматизирани системи.
   2. Предназначени са автоматизация на комуникация и улеснение за намиране на информация.
   3. Създаването му изисква описването на значителните елементи на една страница в семантична форма (съдържание, композиция, предназначение, произход и т.н.).
   4. Пример за това би било страница със снимки на различни цветя, при правилно семантични описани елементи, търсенето на снимки за рози би довел даден потребител в тази страница, ако има снимки на рози и са описани чрез семантични тагове.

Семантичен HTML не е друга версия на HTML. Той е най-обикновен HTML а семантичен го прави начина, по който той бива използван. Семантичен означава „значещ“. По принцип HTML не е нещо което е предназначено за крайния потребител пред екрана, а за разработчиците и машините. Той пояснява вида на съдържанието и указва неговата йерархия. Един сайт може да бъде изграден без новите семантични елементи и за обикновенният потребител това да не е проблем, но как ще се ориентира един незрящ човек или така наречените уеб паяци. За тях всичко ще е еднакво и те няма да разберат кое съдържание е по-важно от друго. Поради тези причини семантичния HTML е много важен за SEO оптимизацията и лесното разбиране на на сайтовете от хора със специални нужди.

Във HTML5 се ползват новите семантични тагове за да се подреди правилно документа.

HTML5 семантични тагове

<nav>

<header>

<main>

<section>

<article>

<aside>

<footer>

<figure> + <figcaption>

<details> + <summary>

<time>

<address>

<dialog>

<audio>

<video>

1. Обяснява и демонстрира начините за създаване на адаптивен (responsive) дизайн.
   1. Използват се модерни методологии за структуриране, като FlexBox или CSS Grids.
   2. Използват се CSS Media Queries за да се определи формата и резолюцията на екрана и се използват различни стилове според това.

Съвременния уеб дизайн трябва да бъде адаптивен, т.е. да е съвместим с различни видове устройства. Представянето на уеб страницататрябва да е еднакво при всички устройства. Това е възможно чрез така наречения отзивчив и адаптивен уеб дизайн. Отзивчивия уеб дизайн (Responsive Web Design - RWD) представлява стилизиране уеб страницата, с цел да улесни взаимодействието с потребителите и промени нейната визуализация към компютър, таблет, мобилен телефон и др. устройства. Този вид уеб дизайн се адаптира спрямо различните резолюции на екрана. По този начин се премахва нуждата от мащабиране на екрана. Прилагането действие от потребителя се свежда до вертикално скролиране, за да се прочете цялото съдържание.

Предимства на отзивчивия дизайн:

 разработва се само един сайт, а не няколко различни според резолюцията на устройството;

 по-добро индексиране от Google;

 удобно и добре изглеждащо съдържание;

 по-голяма достъпност от множество потребители използващи различни устройства.

Недостатъци на отзивчивия дизайн:

 Съдържанието на отзивчивия сайт е възможно да изглежда като накъсани парчета;

 Основен недостатък е скоростта на зареждане;

 Показването на видео и слайдове затрудява нормалното ползване.

Уеб сайтовете създадени чрез отзивчив дизайн (responsive) организират съдържанието към средата на уеб страницата като използват:

 гъвкави пропорционални решетки (Responsive Grid) ;

 отзивчива уеб типография (Responsive Web Typography);

 гъвкави изображения (Responsive Images);

 CSS медийните заявки ( @media ).

Когато към описанието на елементите в CSS се въведат допълнителни свойства и правила за представяне, блоковете в уеб страницата заемат последователна позиция и изпълват видимото пространство на браузъра. Това се реализира с:

1) въвеждане на мета тага viewport;

2) структура на решетката - фиксирана ширина на блоковете в проценти при различните ширини на екрана;

3) медийни заявки -@media за различни ширини на екрана.

Характеристики на VIEWPORT - чрез мета тага viewport може дасе контролира прозореца на представяне на сайта. Метода се въвежда с HTML 5:

<meta name="viewport" content="width=device-width, initial-scale=1.0“>

1) Тагът <meta> с име viewport дава инструкции на браузъра да контролира размерите на уеб страницата и мащабирането.

2) Характеристиката width=device-width, определя ширината на страницата да следва ширината на екрана на устройството.

3) Характеристиката initial-scale=1.0, определя първоначалното мащабиране на страницата когато тя се зарежда в браузъра. Медийните заявки са CSS техника стандартизирана с CSS3. Техниката използва запазената дума media. Тя се съчетава с условие, което ако е вярно се изпълняват описаните свойства към медийната заявка. Пример:

@media тип и списък от заявки (условие) {

група от свойства

}

В Responsive Web Design(RWD) @media се използва, за да се зададе при каква минимална или максимална ширина на страницата да се изобразят елементите от уеб страницата по определен начин. Възможно е условието в @media да се ограничи или разшири с вмъкването на логически оператори: and (логическо и); not (логическо не); изброяване на повече условия разделени със запетая.

Всички HTML елементи могат да бъдат разгледани като кутии. В CSS, „Бокс Модел“ терминът се използва, когато говорим за дизайн и оформление. Като една кутия, която обвива HTML елемента. Тя се състои от margin(разстояние отвън), border(рамка), padding(разстояние отвътре) и content(съдържание). Това ни позволява да слагаме рамки на елементите и да определяме разстоянията между другите елементи.

1. Работа с обекти и събития в JavaScript. Ученикът пише код и модифицира съществуващ такъв спрямо конкретна задача.
2. Демонстрира манипулирането на DOM.

DOM е абривиатура идваща от Document Object Model и се отнася за структурата на страницата и ни дава изчистена конвенция за визуализиране и отнасяне към всяка една част от тази структура. DOM е мулти-платформен и езиково независим приложно-програмен интерфейс, който позволява на програми и скриптове, да получат достъп до съдържанието на обекти в HTML, XHTML и XML-документи. Всяка известна структура на документ с помощта на DOM може да бъде представена в дървовидна структура от взаимовръзки, всяка от която връзки представлява елемент, атрибут, текст, графика, или друг обект. Връзките са свързани помежду си в йерархия „родител-дете“.