Тема 12

Интернет програмиране

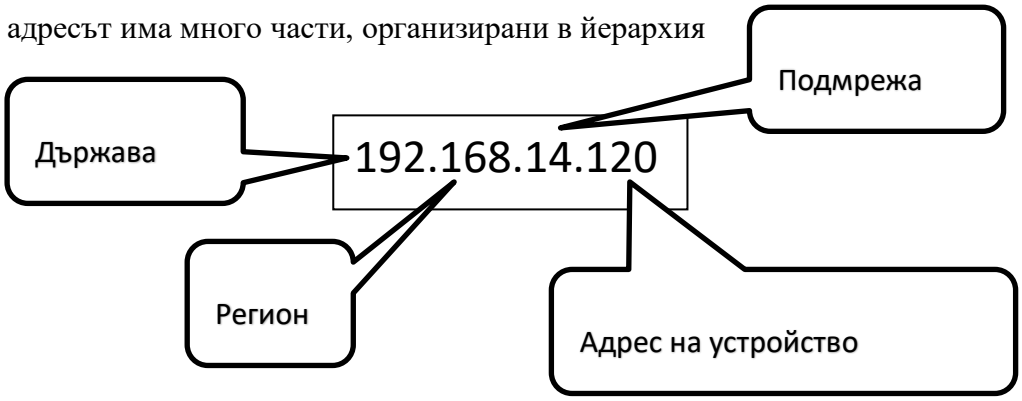
1. Обяснява и диференцира различните протоколи.

Мрежов протокол - Набор от правила и стандарти, които позволяват комуникация между мрежовите устройства. Мрежовите протоколи включват механизми за идентифициране на устройствата и осъществяване на връзка помежду си.

IP(Internet Protocol)- Един от най-важните протоколи, използвани в интернет комуникацията, е Интернет протоколът (IP).

Всички устройства в Интернет имат адреси, тези адреси се наричат IP адреси. IP адресът е уникален за всеки компютър или устройство в края на мрежата.

IP адресът има много части, организирани в йерархия



Тази версия на IP адресиране се нарича IPv4

IPv4 е последователност от четири, трицифрени числа, разделени от точка

Всяко число може да бъде число от нула до 255. IPv4 не е достатъчен за всички мрежови устройства, свързани към интернет. През 1995 г. е създадена нова версия на интернет протокола, наречена IPv6.

IPV6 използва 128 бита. Тези 128 бита са организирани в осем 16-битови части. Всеки 16-битов блок се преобразува в шестнадесетичен и се разделя с двоеточие.

Пример на пълен IPv6 адрес:

3FFE:F200:0234:AB00:0123:4567:8901:ABCD

TCP протокол (Transmission Control Protocol) - осигурява зависим от връзката трафик между две устройства в мрежата. Когато се изпраща съобщение, между компютъра, който изпраща и компютъра получател трябва да се изгради сесия (връзка). Съобщението, което се изпраща се разделя на сегменти, като всеки сегмент има пореден номер. При достигането на сегментите при получателя, те се подреждат по тези номера, за да се получи оригиналното съобщение. Ако някой от сегментите не е

пристигнал се изисква неговото повторно изпращане. *Протоколът TCP осигурява гарантирано достигане на информацията до получателя.*

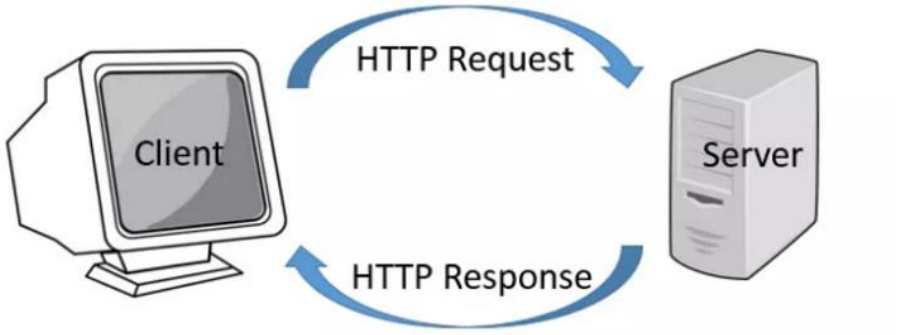
UDP (User Datagram Protocol) протокол - Протоколът UDP осигурява негарантирана доставка на данни.UDP осъществява изпращането на данните и не се занимава с проверка на получаването им. Съобщението, което се предава се нарича дейтаграма (datagram).

Програмите, използващи UDP протокола трябва да реализират самостоятелно:

• препредаване на загубени дейтаграми;
• игнориране на повторения;
• фрагментиране и обединяване на големи потоци данни.

2. Дефинира понятието HTTP заявка, прави изводи за различните HTTP методи и избира метод за конкретна ситуация.

HTTP (**Hyper Text Transfer Protocol**) – мрежов протокол за пренос на хипертекст. В HTTP протокола се използват понятия като клиент (обикновено това са Web-браузърите (или web навигаторите) – т.е. самите приложения и сървър (това са уеб сървърите – т.е. самите приложения).



HTTP дефинира методи за посочване на желаното действие, което трябва да се извърши върху идентифицирания ресурс.

Метод	**Описание**
GET	Извличане / зареждане на ресурс
POST	Създаване / съхраняване на ресурс
PUT	Актуализиране на ресурс
DELETE	Премахване на ресурс

HTTP съобщение за заявка

Съобщение за заявка, изпратено от клиент, се състои от:

- HTTP линия за заявка
  - Метод на заявка (GET / POST / PUT / DELETE / …)
  - URI (URL)
  - Версия на протокола
- HTTP хедъри
  - Допълнителни параметри
- Тяло на HTTP заявка - незадължителни данни, напр. публикувани формулярни полета

```
<method> <resource> HTTP/<version>
<headers>
(empty line)
<body>
```

Get метод на заявка:

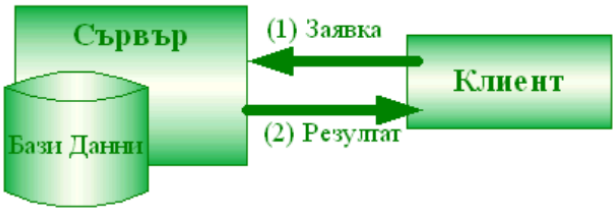
```
<form method="get">
   Name: <input type="text" name="name" />
   Age: <input type="text" name="age" />
   <input type="submit" />
</form>
```

HTTP съобщение за отговор - Съобщението за отговор, изпратено от HTTP сървъра, се състои от:

- HTTP линия на състоянието на отговора представя:
    - Версия на протокола
    - Код на състоянието
    - Текст на състоянието
- Хедърите:
    - Предоставят метаданни за върнатия ресурс
- Тяло на отговора е:
    - Съдържанието на HTTP отговора (данни)

3. Обяснява и представя графично клиент-сървърната комуникация.

Клиент/сървър архитектурата не винаги се ограничава до комуникацията с единствен сървър. Понякога клиентските заявки се разпределят между много сървъри. Но в най-честия случай края на клиент/сървър веригата е сървърът с базите данни, а клиентското приложение се грижи за логиката и графичния интерфейс (вижте фигура 1).



Комуникацията между Web клиента и Web сървъра се осъществява чрез използване на протокола HTTP. **HTTP** (**H**yper **T**ext **T**ransfer **P**rotocol) служи за обмен на документи между сървър и клиент, и е част от протоколния стек TCP/IP за управление на поток от данни в Интернет. Всъщност протокола HTTP функционира на базата на проста схема от тип “въпрос–отговор". Клиентът изпраща заявка към сървъра, на която сървърът отговаря. Архитектурата клиент/сървър има три основни компонента: клиент, сървър и връзката помежду им.

4. Различава смисъла на употребата и необходимостта от HTML, CSS и JavaScript.

- HTML – **H**yper**T**ext **M**arkup **L**anguage
    - Нотация за описание
        - структура на документ
        - форматиране (презентация)
- Таговете предоставят метаинформация за съдържанието на страницата и определят нейната структура
- Един HTML документ се състои от много тагове, които се влагат

HTML терминология:

- Тагове – най-малкият елемент в HTML
- Атрибути – свойсвтвата на таговете - размер, цвят и т.н
- Елементи – комбинация от отварящ, затварящ таг и атрибути

CSS определя стила на HTML елементите

- Определя шрифтове, цветове, полета, размери, позициониране и др.
- CSS се декларира в следния формат: **свойсвто:стойност**

Вграденият CSS дефинира правила за форматиране на определен HTML елемент:

```
<p style="color: red;">I am a RED text paragraph</p>
```

HTML описва текст с форматиране, изображения, таблици, формуляри и т.н.

- CSS добавя стилизиране към HTML документите
  - Шрифт, цвят, фон, подравняване
- Уеб сайтовете се състоят от HTML + CSS + изображения
  - Може да съдържа JavaScript код

JavaScript (JS) е скриптов език

- Изпълняват се команди (скрипт)
  - Не се компилира
- Може да работи в интерактивен режим
- Наред с **HTML** и **CSS**, **JavaScript** е една от **3-те основни технологии** в уеб света
  - JavaScript позволява динамичност и интерактивност в уеб страниците
    - Има достъп до **DOM** и API(известия, геолокация и т.н.)

5. Дефинира и използва коректно HTML тагове.

- Заглавия:

Html има шест различни HTML заглавия

<h1> определя най-важното заглавие.

<h6> определя най-малко важното заглавие.

- Параграфи

Тагът **<p>** дефинира параграф

Тагът **
** дефинира нов ред

- Булети и Номерирани Списъци

Тагът <ul> - за булет списък

Тагът <ol> - за номериран списък

Пример:

```
<ul>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ul>
```

- Хипервръзки:

Създават се с тага <a>

```
<a></a>
```

Адресът се посочва в **href=""** атрибута

```
href="https://abv.bg"
```

Външна хипервръзка

```
<a href="https://abv.bg">Препратка към abv поща.</a>
```

Локални хипервръзки

```
<a href="welcome.html">Показва "welcome.html"</a>
```

- Снимки

Изображенията са външни файлове, които се вмъкнати чрез <img> тага.

```
<img
  src="images/SoftUni-logo.png"
  alt="SoftUni logo (blue)"
  width="400"
  height="313" />
```

- Таблици – дефинира се с тага <table>, в който се посочват редове <tr> и клетки<td>

```
<table>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
</table
```

6. Задава свойства на HTML компонентите чрез CSS.

- **color**: определя цвета на буквите
- **font-family**: трябва да съдържа няколко шрифта. Ако браузърът не поддържа първия, той ще опита следващия
- **font-size**: задава размера
- Блокови елементи **(;<h1>;<p>)**:
  - Винаги започвайте на нов ред
  - Заемат цялата налична ширина
- ** ** елемента:
  - често се използва като контейнер за други HTML елементи
- Вградени елементи **(;<a>;<img>)**:
  - Не започват на нов ред
  - Заемат само толкова ширина, колкото е необходима
- **** елемент:
  - Често е използван за контейнер за текст
- **border**: определя типа, дебелината, цвета
- **border-radius**: закръгля граничните краища
- **background**: задава фона

Външни Отстояния

- Използва се за генериране на пространство около елементи
- **margin** свойсвото задава размера на празното пространство извън границата

Вътршени Отстояния

- Използва се за генериране на пространство около съдържанието
- Свойството **padding** задава размера на празното пространство вътре в границата

CSS Селектори

- **.class** – избира група елементи
  с посочения клас
- **#id** – избира уникален елемент
- **tag** – избира всички посочени тагове
- ***** - избира всичко

Пример:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="styles.css">
  </head>
  <body id="content">
  <p>This is a <span class="special"> special beer</span> for <span class=
"special">special drinkers</span>.</p>
  </body>
</html>
```

style.css

```
#content {
  background: #EEE;
}
p {
  font-size: 24pt;
}
.special {
  font-style: italic;
  font-weight: bold;
  color: blue;
}
```

7. Описва и демонстрира употребата на семантични елементи за създаване на семантична страница.

Семантичният уеб е обединяващо звено на различните стилове, като HTML Markup, CSS правила, JavaScript код, снимки, аудио, видео и други ресурси, като ги съчетава в един общ формат.

Семантични елементи

- Елемент <header> – представлява заглавие на елемент или група от елементи. Тага <header> позволява използването на няколко заглавия в един документ за разлика от възможността която предоставя HTML4.
- Елемент <nav> – дефинира навигацията на сайта. Той е предназначен за големи блокове от навигационни връзки. Не всички връзки трябва да са в <nav> елемента, а само главното навигационно меню.

- Елемент <article> – представлява предмет/статия. Съдържанието в <article> трябва да има смисъл само по себе си и да може да се разпространява независимо и отделно от останалата част на сайта. Използва се най-често за публикация във форум; публикация в блог; новини; коментари и др.
- Елемент </section/> е тематично групирано съдържание, обикновено със заглавие. Този елемент позволява влагане на нова сегмантична структура във вече съществуващ елемент.
- Елемент <aside> – той представя съдържание което е странично от основното съдържание на страницата. Съдържанието в <aside> се счита за неважно и може да бъде пропуснато, ако сайта се възпроизвежда на устройство с малък екран като телефон.
- Елемент <footer> – може да съдържа информация за автора на страницата; навигация на сайта; авторските права върху използваните материали и връзки към други (подобни) публикации.

8. Обяснява и демонстрира начините за създаване на адаптивен (responsive) дизайн.

Отзивчивия уеб дизайн (Responsive Web Design - RWD) представлява стилизирана 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.

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.

Чрез мета тага **viewport** дизайнерите имат възможност да контролират 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.