МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

WEB-ТЕХНОЛОГІЇ

*Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для студентів, які навчаються за спеціальністю
151 "Автоматизація та комп'ютерно-інтегровані технології"*

Київ

КПІ ім. Ігоря Сікорського

2020

Рецензент: *Сірий О.А., к.т.н., доцент*

Відповідальний редактор: *Любицький С.В., ст. викладач*

Гриф надано Методичною радою КПІ імені Ігоря Сікорського (протокол № 9 від 30. 04. 2020 р.) за поданням Вченої ради факультету (протокол № 9 від 29 . 04 .2020 р.)

Електронне мережне навчальне видання

Бунке Олександр Сергійович, к.т.н., доцент

WEB-ТЕХНОЛОГІЇ

WEB-технології [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології», освітньо-професійна програма «Автоматизація та комп'ютерно-інтегровані технології кібер-енергетичних систем» /Укладач: О. С. Бунке ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 1,0 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2020. – 28 с.

Посібник розроблений на підставі робочої програми навчальної дисципліни «WEB-технології» та призначений для проведення лабораторних занять, підвищення розуміння основ WEB-технологій.

Призначений для студентів, які навчаються за освітньою програмою підготовки бакалаврів за спеціальністю 151 "Автоматизація та комп'ютерно-інтегровані технології".

Спрямований на формування у студентів умінь та навичок проектування та моделювання WEB-документів. Забезпечує студентів необхідними теоретичними знаннями для виконання практичних завдань, запланованих впродовж семестру.

ЗМІСТ

ВСТУП

Дисципліна «Web-технології» є базовою частиною професійного циклу дисциплін. Це логічне продовження дисциплін «Інформатика і програмування», «Інформаційні технології», «Програмна інженерія», «Обчислювальні системи, мережі та телекомунікації», «Бази даних». Використання даної дисципліни необхідно як попереднє для курсів професійного циклу («Високорівневі методи інформатики та програмування», «Предметно-орієнтовані ЕІС»), а також для написання Випускної кваліфікаційної роботи.

Метою освоєння дисципліни «Web-технології» є розширення світогляду і формування знань, уявлень і навичок про промислову розробку інформаційних Web-ресурсів.

Основними завданнями дисципліни є:

- формування навичок роботи в мережі з Web-ресурсами та Web-послугами;
- формування уявлення про структуру та принципи функціонування і розробки сучасних Web-ресурсів;
- ознайомлення з основними методами сучасних Web-технологій у професійній діяльності, а також із засобами підтримки прийняття рішень і можливостями їх застосування в задачах управління інформаційними ресурсами підприємства.

В результаті вивчення дисципліни студент повинен:

1) знати:

- поняття Web-технології, Web-послуги та їх класифікацію;
- поняття Web-ресурси та їх класифікацію;
- характеристики основних секторів ринку Web-послуг і питання використання ділової інформації;
- технології створення Web-додатків і інформаційних ресурсів;
- мови клієнтських і серверних сценаріїв;

2) вміти:

- організувати роботу щодо доступу до ділової інформації на базі сучасних Web-технологій;
- проводити аналіз, проектування, якісні показники Web-додатків і інформаційних ресурсів;
- організовувати процес розробки Web-додатків і інформаційних ресурсів, вести документацію відповідно до сучасних стандартів.

ЛАБОРАТОРНА РОБОТА № 1

Розробка статичної верстки WEB-сторінки (основні типи сторінок сайту обраної тематики)

Мета роботи: розробити WEB-сторінку, використовуючи різні елементи форматування вмісту і тексту.

HTML – це мова форматування тексту. HTML-документ – це просто текстовий файл з розширенням * .html.

Фрагмент тексту, укладений між символами <i>, називається тегом (від англійського слова TAG, в перекладі – мітка).

Тег – це символьна конструкція з <(відкриває) і> (закриває) кутових дужок, між якими знаходиться конкретний символ або рядок символів, які веліли браузеру відображення такого змісту документа відповідно до їх призначення.

Мова HTML використовує різні теги, що вводяться в текстові документи, які вказують, яким чином інформація повинна зчитуватися WEB-браузером. Більшість тегів HTML парні, тег відкриває і закриває. Вони охоплюють текст, що позначений ними. Тег пари, що закриває завжди починається з прямої косої риси. Теги цього тексту визначають, яким чином текстова інформація і графіка повинні бути представлені на екрані. Крім тегів <HTML>, <HEAD>, <TITLE> і <BODY>, які відповідно до стандартів HTML і WWW повинні бути присутніми в кожній WEB-сторінці, при введенні тексту HTML-файлу для розбиття його на смислові групи і стильового оформлення тексту використовуються кілька тегів розмітки.

Таблиця 1 – Теги, що задають структуру документа

<!DOCTYPE ..... >	Декларування типу документа	Використовується для вказівки, з яким стандартом HTML сумісний документ

Продовження таблиці 1

<HTML>	Тип структури HTML	Початок структури HTML
<HEAD>	Початок опису документа	Розділ опису документа може включати мітки <TITLE>, <META>, <BASE> і <LINK>
<TITLE> ..... </TITLE>	Ім’я документа	Те, що буде вважатися заголовком (Назвою) документа
<META	Мета-інформація	Служить для вказівки: а) технічної інформації про документ б) інформації про зміст документа
HTTP-EQUIV="ім’я" CONTENT="значення"	Інформація для HTTP-сервера: привласнити будь-яке значення якомусь заголовку	Для використання кирилиці на HTML-сторінках вкажіть в заголовку: <META HTTPEQUIV = "ContentType "CONTENT =" txt / ht ml; charset = Windows-1251 ">

Продовження таблиці 1

NAME=" ім’я " CONTENT="значення">	Завдання мета-змінної і присвоєння їй значення	Завдання мета-інформації про документ. Пошукові служби судять по ній про утримання документа.
<LINK ..... >	Вказівки про гіперзв'язок даного документа	Атрибути, що вказано – такі ж, як у мітки <A> (Якір, anchor). Служить для вказівки перетинів, а також відносин між документами
</HEAD>	Кінець опису документа	
<BODY	Початок документа	Вказуються установки для показу документа
BACKGROUND="URL"	Фонова картинка	В лапках вказується URL картинки (.gif або .jpg)
BGCOLOR="#$$$$$$"	Колір фону	В лапках вказується номер кольору або його назва
TEXT="#$$$$$$"	Колір тексту	
LINK="#$$$$$$"	Колір посилання	
VLINK="#$$$$$$"	Колір посилання, що переглянуто	
ALINK="#$$$$$$" >	Колір активного посилання	

Продовження таблиці 1

</BODY>	Кінець документа	
</HTML>	Кінець структури HTML	

Завдання

Крім використання тегів <HTML>, <HEAD>, <TITLE> і <BODY>, які відповідно до стандартів HTML і WWW повинні бути присутніми в кожній WEB-сторінці, при організації тексту всередині WEB-документа за допомогою списків, а також використанні попереднього форматування і графіки застосовуються відповідні елементи мови HTML з певним набором тегів і необхідних атрибутів.

1. Використовуючи редактор тексту Блокнот (Notepad++) розробіть сторінку, використовуючи всі наведені в таблиці теги мови HTML.

Збережіть сторінку в робочому каталозі. Відформатуйте текст Вашого варіанту завдання згідно вимогам у дужках.

2. Перегляньте на свою сторінку за допомогою встановленого на комп'ютері браузера і добийтеся максимальної подібності в поданні сторінки браузером.

Для виконання даної роботи вам необхідно використовувати наступні теги елементів мови HTML (див. табл. 2).

Таблиця 2 – Основні теги в лабораторній роботі

Приклад елементу	Пояснення
<HTML>...</HTML>; <HEAD>...</HEAD>; <BODY>...</BODY>	Обов’язкові елементи для будь-якої сторінки HTML
<TITLE>...</TITLE>	Назва сторінки

Продовження таблиці 2

<H1>...</H1>; <H2>...</H2>; <H3>...</H3>; <H4>...</H4>; <H5>...</H5>; <H6>...</H6> атрибут align="left|right|center"	Стилі заголовків всередині сторінки
<P>...</P> атрибут align="left|right|center|justify"	Елемент нового абзацу
 	Тег перекладу рядка. Не поділяє два фрагмента порожнім рядком
<HR> атрибути: size, noshade, width, align, color	Тег горизонтальної лінії
<B>...</B>; <I>...</I>; <TT>...</TT>;<S>...</ S>; <U>...</U>;<BIG>...</BIG>; <SMALL>...</ SMALL>; _{...};^{...}	Оформлення тексту
<FONT>...</FONT> атрибути: size, face, color	Завдання розміру, типу шрифту і кольору тексту
<BODY text="Колір">...</BODY>	Спосіб управління кольором тексту
&-послідовності	Відображення спеціальних символів

ЛАБОРАТОРНА РОБОТА № 2

Застосування CSS для оформлення інтерфейсу web-додатку

Мета роботи: ввести поняття каскадних таблиць стилів, використати їх переваги.

У зв’язку з тим, що HTML не мав можливості управління зовнішнім виглядом Веб-сторінок, щоб вирішити ці проблеми, консорціумом W3C була розроблена технологія Cascading Style Sheets або CSS (Каскадні таблиці стилів) - технологія опису зовнішнього вигляду документа, написана мовою розмітки. В CSS надається можливість призначити усім об'єктам стиль, опис якого може зберігатися в самому HTML-файлі, або в окремому файлі.

ПРАВИЛА В CSS. CSS надає можливість створювати правила, які легко змінювати, редагувати і застосовувати до усіх визначених нами елементів. Кожне правило, складається з двох частин. У лівій частині міститься *селектор (selector),* а у правій - *блок визначення (declaration block).* Блок визначення складається з набору *властивостей (property)* та їх *значень (value).*

Селектор – елемент, який визначає правило.

Властивість описує елемент, що вводиться.

Значення визначають природу властивостей.

Приклад: **h1** *(селектор)* **{color: red}** *(блок-значення)*

ДОДАВАННЯ ТАБЛИЦЬ СТИЛІВ В HTML-ДОКУМЕНТ. Існують **три способи** додавання правил CSS в HTML-документи:

- ***<u>СПОСІБ ПЕРШИЙ</u>: ДОДАВАННЯ CSS В HTML-ТЕГ.***

У цьому способі CSS додається в HTML-документ за допомогою HTML-атрибуту style у середині будь-якого HTML-тегу, що знаходиться у контейнері <body>.

```
<html>
  <head>
    <title>Sample BG color</title>
  </head>
  <body style="background-color: #FF0000;">
    <p>You see red color</p>
  </body>
</html>
```

Цей спосіб використовується у тому разі коли окремому елементу потрібно надати декілька стилів не використовуючи вбудовані або зовнішні стилі.

Застосування цього способу несе за собою певні недоліки:

- збільшується об’єм файлу, що приводить до збільшення часу завантаження веб-сторінки;

- ускладнює редагування документів.

- ***СПОСІБ ДРУГИЙ: ВСТАНОВЛЕННЯ СТИЛЮ ДЛЯ ТЕГІВ HEAD БЛОЦІ***

CSS додається в HTML-документ за допомогою HTML-тегу <style> в середині контейнеру <head>. В ньому описуються всі стилі, що будуть використані.

```
<html>
  <head>
    <title>Sample head</title>
    <style type="text/css">
      body {background-color: #FF0000;}
```

```
    </style>
    <p>You see red color</p>
  </body>
</html>
```

- ***<u>СПОСІБ ТРЕТІЙ</u>: ПОСИЛАННЯ НА ТАБЛИЦЮ СТИЛІВ***

Зовнішня таблиця стилів являє собою звичайний текстовий файл з розширенням css.

Для того щоб зробити посилання на зовнішній файл із HTML-документа (index.htm) на файл таблиці стилів (style.css) треба у контейнері <head>вставити наступну стрічку:

```
<link rel="stylesheet" type="text/css" href="style/style.css" />
```

Це посилання указує браузеру, що він повинен використовувати правила відображення HTML-файлу з CSS-файлу.

```
<html>
  <head>
    <title>Sample of CSS link</title>
  <link rel="stylesheet" type="text/css" href="style/style.css" />
  </head>
  <body>
  </body>
</html>
```

Найважливішим тут є те, що один CSS-файл можна використовувати для управління відображення багатьох HTML-документів.

Управління шрифтом. В старих версіях HTML шрифт оформлявся за допомогою тегу <font> але цей тег треба було додавати кожного разу, коли було необхідно встановити шрифт, від цього збільшувався розмір веб-сторінки, незручно було змінювати властивість шрифту.

Щоб задати сімейство шрифтів використовується властивість font-family. В цій властивості завжди вказується ряд шрифтів, розділених комою, наприкінці списку вказується сімейство шрифтів. При застосуванні шрифтів до веб-сторінки завжди задається основний шрифт, а потім альтернативний. У разі відсутності на комп'ютері користувача заданих шрифтів то веб-сторінка, як мінімум буде відображена шрифтом, що входить до цього сімейства.

Таблиця 3 – Основні п'ять сімейств шрифтів

Шрифт	**Сімейство шрифтів**
Times New Roman Georgia	serif(з засічками)
Arial Candara	sans-serif (без засічок)
Courier Courier New	monospace (моноширинний)
Bradley Hand ITC Edwardian Script ITC	cursive (рукописний)
Luminari, fantasy	fantasy (декоративний)

Шрифти сімейства serif найкраще підходять для основного тексту сторінки. Засічки допомагають направляти увагу читача уздовж рядка

Шрифти сімейства sans-serif використовуються для оформлення заголовків, панелей посилань та посилань. Шрифти без засічок звертають на себе більше уваги, але погано підходять для довгого читання.

У моноширинних шрифтах усі символи мають однакову ширину. Слід зазначити, що у моноширинних шрифтах збільшується між символьний інтервал, як зліва так і з права символу (цей інтервал є невід'ємною частиною символу). Моноширинні шрифти допускаються тільки для створення якихось особливих ефектів оформлення — наприклад, у даній роботі моноширинним шрифтом набрано фрагменти коду HTML та CSS. У моноширинному шрифті всякий символ має одну і ту ж ширину.

У файлі style.css створимо для оформлення абзацу таке правило:

```
p{font-family: Arial, Candara, Century-Gothic, sans-serif}
```

```
<html>
  <head>
  <title>Sample Sans</title>
  <link rel="stylesheet" type="text/css" href="style/style.css" />
  </head>
  <body>
    <p>Sans-serif text font</p>
  </body>
</html>
```

У цьому прикладі основним шрифтом абзацу є шрифт Arial у разі його відсутності буде загружено наступний, у разі відсутності усіх шрифтів абзац буде оформлено шрифтом сімейства sans-serif.

Властивість font-style визначає стиль шрифту з обраного сімейства може мати наступні значення:

- normal – звичайний шрифт;
- italic – курсивний шрифт (більш декоративний шрифт з нахилом в право);
- oblique – нахилений шрифт (звичайний шрифт нахилений в право).

У файлі style.css створимо таке правило:

```
h1{
   font-family: Arial, Candara, Century-Gothic, sans-serif;
   font-style: normal;
}
h2{
   font-family: Times New Roman, Georgia, serif;
   font-style: italic;
}
```

```
p{
   font-family: Times New Roman, Georgia, serif;
   font-style: oblique;
}


<html>
  <head>
  <title>Tag css sample</title>
  <link rel="stylesheet" type="text/css" href="style/style.css" />
  </head>
  <body>
  <h1>Шрифт цього заголовку звичайний</h1>
  <h2>Шрифт цього заголовку курсив</h2>
    <p> Шрифт цього абзацу нахилено вправо</p>
  </body>
</html>
```

ПОСИЛАННЯ

Псевдокласи. Властивості гіперпосилання засобами CSS можна визначати по-різному, залежно від того, відвідали вже посилання, чи активне воно, чи знаходиться покажчик миші над посиланням. Це дозволяє додати цікаві ефекти на ваш веб-сайт. Щоб оформити гіперпосилання засобами CSS, треба використовувати так звані псевдокласи.

Псевдоклас дозволяє враховувати різні стани або події при визначенні властивостей html-тега. У гіперпосилання є декілька станів та подій.

Псевдоклас : link використовується для посилань на сторінки, які користувач ще не відвідував.

```
a: link{
color: #0000ff;
font-weight: normal;
}
```

Псевдоклас: active використовується для активних посилань.

```
a: active {
```

```
color: #00bfff;
background-color:ffd700;
{
```

У цьому прикладі у посилання буде змінено колір шрифту та колір фону.

Псевдоклас: visited використовується для посилань на сторінки, які відвідав користувач.

```
a: visited {
color: #0000ff;
font-weight: 400;
}
```

Псевдоклас: hover використовується для посилань, над котрими находиться вказівник миші.

```
a: hover{
font-weight: normal;
text-trasform: uppercase;
letter-spacing: 10px;
}
```

У цьому прикладі при шрифт посилання буде нормальної товщини, текст буде відображатися прописними літерами, відстань між літерами буде складати 10px.

Видалення підкреслювання посилань. Видалити підкреслювання посилань дуже просто. Для видалення підкреслювання достатньо встановити властивість text-decoration зі значенням none.

```
a: link{
color: #0000ff;
font-weight: normal;
text-decoration: none;
}
```

Ідентифікація і групування елементів. Атрибут class вказує, що елемент є членом певного класу. Для прикладу візьмемо документ в якому є два списки посилань, але треба щоб кольори посилань цих списків відрізнялися.

```
<p>Список №1:</p>
<ul>
<li><a href="link1_1.htm">Посилання 1,1</a></li>
<li><a href="link1_2.htm">Посилання 1,2</a></li>
<li><a href="link1_3.htm">Посилання 1,3</a></li>
</ul>
<p>Список №2:</p>
<ul>
<li><a href="link2_1.htm">Посилання 2,1</a></li>
<li><a href="link2_2.htm">Посилання 2,2</a></li>
<li><a href="link2_3.htm">Посилання 2,3</a></li>
</ul>
<a href="index.htm">На головну</a>
```

Щоб досягти нашої мети розділимо посилання на дві категорії за допомогою привласнення класу кожному посиланню атрибутом class.

У файлі з html-кодом додамо class="classname" в тег до якого ми будемо застосовувати клас.

```
<p>Список №1:</p>
<ul>
<li><a href="link1_1.htm" class="list1">Посилання 1,1</a></li>
<li><a href="link1_2.htm" class="list1">Посилання 1,2</a></li>
<li><a href="link1_3.htm" class="list1">Посилання 1,3</a></li>
</ul>
<p>Список №2:</p>
<ul>
<li><a href="link2_1.htm" class="list2">Посилання 2,1</a></li>
<li><a href="link2_2.htm" class="list2">Посилання 2,2</a></li>
<li><a href="link2_3.htm" class="list2">Посилання 2,3</a></li>
</ul>
<a href="index.htm">На головну</a>
```

У файлі style.css опишемо властивості класів для посилань. Синтаксис опису класу має наступний вигляд:

[selector].[className] {property: value}

```
a {
color: red;
}
a.list1 {
color: #FFCC00;
}
a.list2 {
color: #700000;
}
```

Окрім групування елементів вам може знадобитися ідентифікувати один унікальний елемент. Це можна реалізувати за допомогою атрибуту id.

Особливість id в тому, що в документі не може бути більш за один елемент з даним конкретним id. Кожен id має бути унікальним. У інших випадках треба використовувати атрибут class.

Завдання

1. Створіть новий HTML документ, що містить основний заголовок H1, Таблицю із двох колонок, в лівій — ненумерований список, в правій заголовок H2 та абзац з текстом (рис. 1).

2. Створити набір CSS стилів, розташувавги його у HEAD, оформити всі елементи сторінки на свій смак. Лівий список зробити посиланнями, що реагують на наведення миші та змінюють колір (застосувати :hover)

3. Перенесить створені стилі у окрейми файл css/main.css та підключіть його за допомогою <link> до документу.

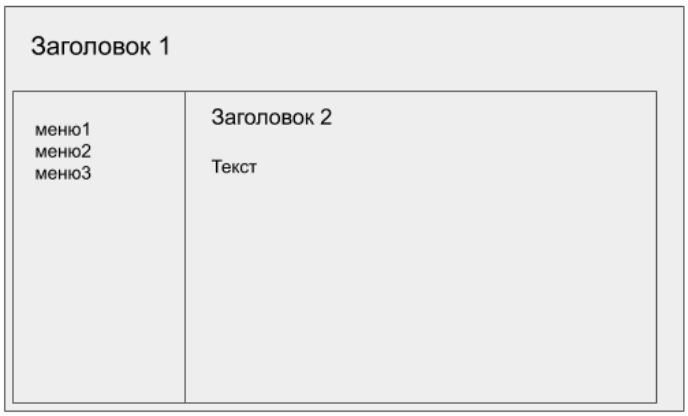


Рис. 1. Шаблон HTML-документу

4. Створіть аналогічний документ з іншим заголовок та текстом, підключіть до нього той самий файл стилів, та впевніться в його роботі.

5. Зробіть висновки щодо зручності використання CSS.

ЛАБОРАТОРНА РОБОТА № 3

Застосування Javascript для динаміки інтерфейсу та валідації форм на сторінці

Мета роботи: ввести поняття сценарію, теги інтерактивності

Сценарії

Мова HTML служить для опису структури тексту, мова CSS - для опису того, як ця структура повинна відображатися браузером. Ці дві мови не передбачають можливості обміну інформацією з користувачем. Однак часто це буває необхідно.

Наприклад, при створенні навчального сайту є потреба з обміном інформацією веб-сторінкою, що реалізує тестування.

Існує мова, звана мовою сценаріїв, яка дозволяє реалізовувати найпростіші дії над даними, які вводить користувач, а також над властивостями елементів веб-сторінки. У нашому випадку ця мова називається JScript.

Модель веб-сторінки, в якій всі елементи утворюють ієрархію об'єктів, де на вершині знаходиться об'єкт window, називається Document Object Model (DOM). У кожного об'єкта є складові, до яких можна звертатися за допомогою символу "точка" (Перш за все це властивості):

об'єкт.властивість

Приклад: звернення до властивості в команді присвоювання

window.status = 'Це рядок стану'

Ця команда ("записати в рядок стану вікна даний текст") могла б працювати в будь-якій певній ситуації, наприклад, при наведенні покажчика миші на той чи інший елемент веб-сторінки. Подібні ситуації називаються подіями. Сценарій - це одна або кілька команд, які запускаються при настанні події.

Розглянемо типовий випадок, коли в якості властивості розглядається стиль деякого елемента. Для звернення до об'єкта (Елементу P) використовується слово this.

Подією буде наведення миші: onmouseover

```
<P style = "color: green"
      onmouseover = "this.style.color = 'red'"> Інтерактивний
абзац </ p>
```

Ми можемо бачити, що колір тексту абзацу при наведенні на текст покажчика миші змінюється на червоний. Щоб спрацювала зворотна ситуація (колір змінювався знову на зелений), потрібно внести сценарій в опис події onmouseout:

```
<P style = "color: green"
       onmouseover = "this.style.color = 'red'"
       onmouseout = "this.style.color = 'green'"> Інтерактивний
абзац </ p>
```

Обробка даних користувача

Розглянемо два способи отримання інформації від користувача:

- введення тексту;
- підтвердження клацанням.

Для прийому тексту від користувача призначений елемент

```
<Input type = "text" name = "ім'я" />
```

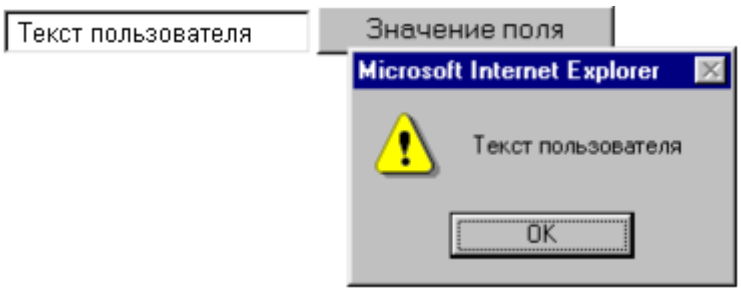
Для створення кнопки призначений елемент

```
<Input type = "button" value = "Напис" onclick = "сценарій" />
```

Скористаємося командою alert (текст) для виведення на екран вікна з повідомленням:

```
<Input type = "text" name = "поле" />
<Input type = "button" value = "Значення поля" onclick = "alert
      (Поле.value) "/>
```

після клацання по кнопці з'являється текст, який Ви самі ввели



Застосування сценаріїв

Точно так же, як замість повторень одного і того ж вмісту атрибута style користуються одноразовим записом стилів у зовнішньому файлі або в елементі <style> </ style>, замість запису безпосередньо в описі події можна посилатися на сценарій, який розміщений в елементі <script> </ script>.

Як і <style> </ style>, він розміщується в елементі <head> </ head>.

Найчастіше сценарій з'являється там у вигляді функції.

```
function імя_функції (нуль або більше аргументів) {
        команди сценарію
}
```

Розглянемо приклад:

```
<script>
    function вітання (кого) {
       alert ( "Привіт," + кого + "!")
    }
</ script>
...
<input type = "text" name = "хто" />
<input type = "button" value = "Вітатися!"
      onclick = "вітання (хто.value)" />
```

Ми почали обговорення сценаріїв із згадки тестів на веб-сторінці. Щоб створити примітивний тест, нам знадобиться ще один елемент HTML, який приймає введення – радіоперемикач – і конструкція мови JScript, що виконує одну або іншу дію в залежності від умови – умовний оператор. Якщо обрано правильний варіант відповіді, сценарій виводить позитивну реакцію, інакше – негативну.

```
if (умова) команда1; else команда2
```

Якщо умова істинна, то буде виконана команда1,

інакше буде виконана команда2.

Приклад:

В якому році закладено м. Київ?

```
<br/>
<Input type = "radio" name = "питання1" /> 1812 <br/>
<Input type = "radio" name = "питання1" /> 1703 <br/>
<Input type = "radio" name = "питання1" /> 1624 <br/>
<Input type = "button" value = "Перевірити!"
 onclick = "if (вопрос1 [1] .checked) alert ( 'Вірно!'); else
alert ( 'Немає вірної відповіді!') "/>
```

В даному прикладі умова – це обрання (checked) одного з радіоперемикачів. Всього є три перемикача, у яких одне ім'я (Питання1). Доступ до них здійснюється за допомогою індексу (номера в квадратних дужках, який починається з нуля – Питання1 [0], Питання 1 [1] і Питання 1 [2]).

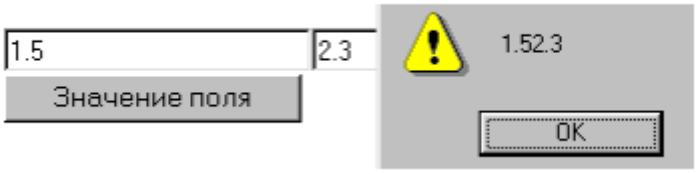
Якщо питань два і більше, зручніше створити одну на всіх функцію перевірки. Вона могла б виглядати наступним чином:

```
function перевірка (вірний) {
 if (верний.checked) alert ( 'Вірно!'); else alert ( 'Ні вірної відповіді!')
}
...
<Input type = "button" value = "Перевірити!"
onclick = "перевірка (Питання1 [1])" />
```

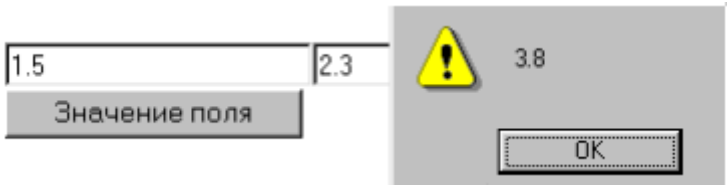
Завдання

1. Створіть функцію, яку можна підключати як до події onmouseover, так і до події onmouseout, яка змінює колір кордону або на колір фону, або на контрастний. Тоді при наведенні миші на елемент навколо нього буде вимальовуватися межа, а при видаленні – зникати.

2. Нехай є два текстових поля: число1 і число2. Команда alert (число1.value + число2.value) не даватиме суму, якщо в полях введені два числа.



Щоб вміст поля трактувався як число, потрібно піддати його перетворенню: parseFloat (число1.value)



Створіть калькулятор, який виконує чотири арифметичних дії (+, -, * і /). Реалізуйте для функції, що відповідає за розподіл, перевірку ділення на нуль.

3. Створіть тест з п'яти питань.

3 *. Якщо в елементі Script помістити команду **var ім'я**, то в пам'яті буде створення змінна *ім'я*. Створіть функцію, в якій значення змінної збільшується на 1 (ім'я = ім'я + 1 або ім'я ++) в разі правильної відповіді, а потім видається загальне число правильних відповідей. Відповідно, в цьому тесті буде не п'ять кнопок після кожного питання, а одна після всього тесту.

Додавання:

1. Якщо в блоці розгалуження потрібно використовувати кілька команд, застосовується форма:

```
if (умова) {
блок 1
} Else {
блок 2
 }
```

2. При необхідності використовувати цикл застосовують конструкцію for:

```
 for (змінна = значення; умова; операція) {
тіло циклу
}
```

Наприклад, підрахуємо суму чисел 1..10:

```
s = 0;
for (i = 1; i <11; i ++) {
s + = I;
alert (i);
}
```

КОНТРОЛЬНІ ПИТАННЯ

1) Основи HTML (синтаксис, структура документа).
2) Основні теги HTML, робота з текстом, списки.
3) Створення посилань.
4) Зображення та їх властивості в HTML.
5) Створення таблиць, відступи, вирівнювання.
6) Використання форм в HTML-документі.
7) Поняття каскадних таблиць стилів CSS, їх призначення.
8) Типи стилів. Переваги стилів.
9) Способи додавання стилів на сторінку.
10) Базовий синтаксис CSS.
11) Класи у CSS.
12) Ідентифікатори у CSS.
13) Селектори і їх види.
14) Можливості JavaScript, де застосовується.
15) Робота JavaScript з DOM сторінки.
16) JavaScript команди і коментарі.
17) Змінні в JavaScript і операції над ними.
18) Арифметичні оператори в JavaScript.
19) Логічні оператори та умовні конструкції в JavaScript.
20) Вікна оповіщення і підтвердження в JavaScript.
21) JavaScript функції.
22) Локальні і глобальні змінні.
23) Цикли JavaScript.
24) Події та їх обробка.
25) Перевірка форм в JavaScript.
26) Спеціальні символи в JavaScript.
27) Методи об'єктів в JavaScript.
28) Масиви в JavaScript.

НАВЧАЛЬНО-МЕТОДИЧНІ МАТЕРІАЛИ

1. Вайк Аллен и др. JavaScript. Энциклопедия пользователя: Пер. с англ. – К.: ООО «ТИД ДС», 2001. – 480с.

2. Лясин, Д.Н., Абрамова О.Ф. Динамическое формирование HTML– документов на стороне сервера: методические указания / Д.Н. Лясин; ВПИ (филиал) ВолгГТУ. – Волгоград, 2018. – 29 с.

3. Вайнман Л., Вайнман В. Динамический HTML. Руководство разработчика Web-сайтов. – К.: ООО «ТИД ДС», 2001. – 464с.

4. О.Ю. Ніколаєнко, С.М. Кравченко, С.А. Вернигоренко. Використання HTML та JavaScript для створення WEB-документів – Київ 2003.

5. Использование HTML 4.0, 4-е издание: Пер. с англ./М. Браун, Дж. Хоникатт и др. – К.:Издат.дом”Вильямс”, 2000. – 784 с.

6. Кирсанов Д. Веб-дизайн: книга Дмитрия Кирсанова. – СПб.: Символ-Плюс, 2001. – 376с.

7. Матросов А.В., Сергеев А.О., Чаунин М.П. HTML 4.0. – СПб.:БХВ-Петербург, 2000. – 672 с.

8. Ніколаєнко О.Ю. Використання НІТ у курсі “Технології створення Web-вузлів”//Вісник.-К: НПУ імені М.П. Драгоманова, 2002. – Випуск 3. – С.76-78

9. Чак Мусчиано. HTML и XHTML. Полное руководство. – СПб.: Символ-Плюс, 2001. – 700с.