ОСНОВНІ ПОНЯТТЯ МОВИ JAVASCRIPT

План:

1. Складові частини JavaScript.
2. Ключові поняття JavaScript: об’єкт, властивості, методи.
3. Елементи мови JavaScript: константа, змінна, функція, оператор, вираження. Ключові слова.
4. Розміщення програми на мові JavaScript.
5. Порядок запуску скриптів.

Складові частини JavaScript.

JavaScript - це мова програмування, яка дозволяє створювати взаємодію з веб-сторінками. Вона використовується для розробки динамічного контенту на веб-сторінках. JavaScript базується на стандарті ECMAScript - це опис мови, який визначений в стандарті ECMA-262.

1. JavaScript складається з трьох основних частин:
2. Ядро (ECMAScript) - це базова функціональність мови JavaScript.
3. Об'єктна модель документа (Document Object Model, DOM) - це засоби для роботи з веб-контентом, які дозволяють змінювати вміст і структуру веб-сторінки.
4. Об'єктна модель браузера (Browser Object Model, BOM) - це засоби для взаємодії з браузером, такі як можливість відкривати нові вікна, керування історією браузера та інші.

Ключові поняття JavaScript: об’єкт, властивості, методи.

Клієнтська JavaScript

У браузерах по замовчуванню є програма з назвою інтерпретатор JavaScript, яка дозволяє браузеру виконувати код, написаний на мові JavaScript. JavaScript є клієнтською мовою, що означає, що код виконується на комп'ютері користувача в браузері, а не на веб-сервері.

JavaScript має багато можливостей, таких як додавання анімації та реагування на події, такі як переміщення миші та натискання клавіш. JavaScript також дозволяє перевіряти введені дані у форми перед їх відправкою на сервер, що зменшує навантаження на сервер. За допомогою JavaScript можна створювати та зчитувати cookies, витягувати дані про комп'ютер відвідувача, визначати браузер та змінювати вміст HTML-елементів.

Однак, є деякі обмеження JavaScript. Наприклад, він не може закривати вікна або вкладки, які не були відкриті за його допомогою, не може захистити вихідний код сторінки або заборонити копіювання тексту та зображень зі сторінки. JavaScript також не може здійснювати запити на веб-сторінки, які знаходяться на іншому домені, навіть якщо

вони відображаються в одному браузері. Це гарантує безпеку приватної інформації. Крім того, JavaScript не має доступу до файлів на комп'ютері користувача, за винятком cookies.

JavaScript – це мова скриптів, яка дозволяє створювати програми, які можуть розширювати можливості веб-сторінок.

Скрипт – це програма, яка може бути виконана браузером одночасно з веб-сторінкою.

Подія – це дія користувача або операційної системи, яка може запустити скрипт. Наприклад, натискання клавіші на клавіатурі, клацання мишею, відкриття вікна браузера або спрацювання таймера.

Об'єкт – це контейнер, який містить дані. Він може мати властивості (які описують об'єкт) та методи (які дозволяють змінювати ці властивості). Об'єктами є, наприклад, вікно браузера (window), веб-сторінка (document), фрейм (frame), текстове поле (text), кнопка (button). Об'єкти можуть бути пов'язані в ієрархію, наприклад, веб-сторінка може містити форму, яка може мати текстове поле.

Властивості – це змінні, які описують об'єкт. Наприклад, ширина та висота вікна браузера.

Методи – це функції, які дозволяють змінювати властивості об'єкту. Назви методів часто є дієсловами, які описують дії, які можна виконати з об'єктом. Наприклад, open() (відкрити), close() (закрити), write() (написати).

Приклади методів:

Метод alert()

Цей метод генерує діалогове вікно-попередження, що відображає текст, заданий як параметр методу. Єдина кнопка ОК, напис якої не можна змінити, призначена для того, щоб користувач міг підтвердити, що він прочитав попередження.

Метод confirm()

У діалоговому вікні, яке відображує цей метод, є дві кнопки та текст, заданий як параметр методу. Для більшості версій браузерів і платформ це кнопки ОК і Cancel (Скасувати). Таке вікно називають діалоговим вікном підтвердження.

Метод Confirm () повертає значення true (так), якщо користувач клацає кнопку ОК, і false (ні) — якщо кнопку Cancel (Скасувати). Це діалогове вікно і значення, яке воно повертає, можна використовувати для надання користувачу можливості керувати подальшими діями сценарію.

Метод prompt()

Цей метод генерує діалогове вікно запиту. Воно містить повідомлення задане як перший параметр методу, і текстове поле для введення відповіді з підказкою, заданою другим параметром. Дві кнопки, наявні в діалоговому вікні, — ОК і Cancel – дають змогу користувачу закрити діалогове вікно, повернувши у сценарій значення текстового поля (кнопкою ОК) або спеціальне значення null (кнопкою Cancel).

З інформацією, отриманою внаслідок виконання всіх цих методів, можуть далі працювати сценарії: наприклад обробляти відповіді користувача, перевіряти їх правильність, долучати їх до баз даних тощо.

Програмістами створено велику кількість оригінальних скриптів (програм-сценаріїв обробки будь-яких подій з подальшою зміною властивостей об'єктів). Скрипти дозволяють створювати різноманітні

ефекти і виконувати різні дії, наприклад: змінювати колір і форму кнопки при наведенні на неї курсора, визначати версію браузера клієнта, виводити на екран місцевий час змінювати колір фону сторінки і т. д. При створенні сценарію у програмістів є великий простір для творчості. Можна вибирати різні події, від яких відбувається запуск скрипта. Програміст самостійно визначає на який об'єкт будуть спрямовані заплановані ним дії. Слід звернути увагу на той факт, що мова JavaScript має засоби для розв'язання математичних і логічних завдань, виконання символьних операцій, роботи з датами і часом.

Мова JavaScript відрізняється від інших мов, таких як алгоритмічні мови, наприклад, C або Pascal. У програмах, написаних на алгоритмічних мовах, програма починає роботу від моменту запуску і працює безперервно до моменту завершення. У JavaScript скриптах можна створювати програми, які працюють автономно і незалежно одна від одної. Кожна програма запускається при певних діях користувача і припиняє роботу, якщо користувач не взаємодіє з нею. Тому цю мову іноді називають інтерактивною мовою програмування.

У мові JavaScript є кілька основних елементів:

- константи;
- змінні;
- функції;
- оператори;
- вирази.

Константи - це значення, які не змінюються протягом роботи скрипта. В JavaScript існує кілька типів констант: цілі числа (десяткові,

вісімкові та шістнадцяткові), числа з плаваючою точкою, рядкові значення та булеві (логічні) значення.

Змінні - це значення, які можуть змінюватися протягом роботи скрипта. У JavaScript змінні можна уявити собі як комірки оперативної пам'яті, вміст яких може змінюватися протягом виконання програми. Програміст може дати тимчасові імена деяким коміркам пам'яті.

Змінні можуть бути *глобальними* і *локальними*. Локальні змінні діють тільки всередині функції. Глобальні змінні діють в рамках всієї складеної програми. Однак, і глобальні змінні діють лише в межах однієї Web-сторінки.

Імена змінних повинні починатися з букви або з символу підкреслення. В іменах допустимо використання арабських цифр. Імена не можуть містити пробілів. Вони не повинні збігатися з ключовими (зарезервованими словами мови JavaScript).

Так само як і константи, змінні можуть бути цілими, з плаваючою точкою, рядковими і булевими. Перед використанням змінні повинні бути оголошені. Робиться це так:

```
var = vol_sound;
var y12 = 3.123;
var x = "Alisa";
```

У наведеному прикладі оголошено три змінні. Причому для змінних y12 і x відразу визначено їх тип (відповідно з плаваючою точкою і символьна змінна).

Числа позначають самі себе: 1, 12, 145, а ось рядки потрібно брати в лапки (одинарні або подвійні - без різниці):

```
'рядок', "рядок"; //це приклади рядків
```

У JavaScript при оголошенні змінної обов'язково має бути написано ключове слово var.

```
var a; // тут ми оголосили змінну
var a, a1, isVar, is_var; // тут ми оголосили групу змінних
```

Зверніть увагу: в коді JavaScript, так само, як і в HTML і CSS, можна залишати коментарі. Вони можуть бути багаторядковими і однорядковими:

```
var a = 4; //це приклад однорядкового коментаря.
/*
  Це приклад
  багаторядкового коментаря.
*/
var a = 4;
```

Коментарі – це позначки в коді, які браузер не виконує. Їх можна використовувати для пояснення коду, або щоб тимчасово вимкнути код, коли це потрібно. Змінні можна згрупувати в масиви.

Масив – це набір змінних, які мають порядкові номери, називаються індексами.

Функція – це частинка коду, яка виконує певне завдання. Кожна функція має унікальне ім'я і може бути викликана багато разів у різних місцях програми.

Оператори – це конструкції, які визначають дії, що виконуються над константами, змінними та функціями. В JavaScript є різні типи операторів:

1. Оператор присвоєння, який використовується для присвоєння значення змінній;

2. Оператор порівняння, який використовується для порівняння двох значень;
3. Арифметичні оператори, які використовуються для виконання математичних операцій;
4. Оператори інкременту та декременту, які використовуються для збільшення або зменшення значення змінної;
5. Бітові оператори, які виконують операції з бітами чисел;
6. Логічні оператори, які використовуються для визначення правильності умови;
7. Оператори для роботи зі строками, які використовуються для з'єднання та розбиття рядків;
8. Спеціальні оператори, такі як typeof, які використовуються для визначення типу даних змінної;
9. Керуючі оператори, такі як break та continue, які використовуються для керування потоком програми;

Оператор	Назва
+	Додавання
–	Віднімання
*	Множення
/	Ділення
%	Ділення по модулю (повертає залишок від ділення)
++	Збільшення на 1
--	Зменшення на 1

Вирази – набір констант, змінних, функцій, з'єднаних операторами. Кожен вираз в мові JavaScript має закінчуватися точкою з комою.

Код з операцією присвоювання:

```
var a = 2;
a = a + 3;
```

В даному випадку привласнюємо змінної a її поточне значення, збільшене на 3. Однак JavaScript дозволяє записати цей код ще коротше за допомогою оператора +=:

```
var a = 1;
a += 3; // цей код повністю еквівалентний коду a = a + 3;
```

Крім того, існують оператори -=, *=, /=, які еквівалентні наступному коду:

```
var a = 2;
a -= 3; // цей код повністю еквівалентний коду a = a - 3;

var a = 2;
a *= 3; // цей код повністю еквівалентний коду a = a * 3;

var a = 2;
a /= 3; // цей код повністю еквівалентний коду a = a / 3;
```

В JavaScript між числами можна здійснювати різні математичні операції

```
alert(2 + 3); //виведе 5
alert(5 - 1); //виведе 4
alert(2 * 3); //виведе 6
alert(6 / 2); //виведе 3
```

В JavaScript можна отримати доступ до певного символу рядка за його номером таким чином: a[n] – n-ий символ рядка (врахуйте, що нумерація йде з нуля)

```
var a, b; // оголосимо наші змінні
```

```
a = 'abcde'; // у змінній a буде зберігається значення 'abcde'
b = a[0]; // у змінній b буде 'a'
b = a[1]; //в змінній b будет 'b'
b = a[4]; //в змінній b будет 'e'
```

Операція a++ або ++a – збільшує змінну a на одиницю (*інкремент*).

Операція a-- або --a – зменшує змінну a на одиницю (*декремент*).

Приклади:

```
var a = 1;
a++; // збільшить a на 1, що відповідає коду a = a + 1;
alert(a); //виведе 2
var a = 1;
a--; // зменшить a на 1, що відповідає коду a = a - 1;
alert(a); //виведе 0
```

Різниця між ++a и a++.

Хай є код alert(++a) і код alert(a++).

У першому випадку змінна спочатку збільшиться на одиницю, а потім виведеться, а в другому випадку - спочатку виведеться, а потім збільшиться.

В JavaScript, як і в інших мовах програмування, існують ключові слова для деяких спеціальних значень: *undefined, null, true, false, NaN, Infinity, -Infinity*.

Значення *undefined* позначає невизначеність. Наприклад, якщо ми спробуємо звернутися до змінної, якої ми ще не задали значення – то її значення і буде *undefined.*

```
var a;
alert(a); //виведе undefined
```

Значення *null* позначає 'нічого'. Наприклад, можна присвоїти змінної значення *null* в знак того, що там нічого не лежить.

Це значення дуже схоже на *undefined*, відмінність в тому, що *undefined* - це не певне значення, а *null* - певне - нічого

Значення *true* і *false* позначають істину і неправду відповідно. Вони використовуються для таких речей, які передбачають два варіанти відповіді – так чи ні.

Наприклад, на питання 'вам вже є 18 років?' ви можете відповісти так, тобто *true*, чи ні, тобто *false*

Значення *NaN* (*Not-A-Number*) позначає не число. Воно може вийде, наприклад, в такому випадку - коли множиться рядок з буквами на число

```
alert('abc'*3); //виведе NaN
```

Значення *Infinity* і *-Infinity* позначають відповідно нескінченність і мінус нескінченність. Вони виходять, якщо якесь число поділити на нуль – в цьому випадку JavaScript не видає помилку, як в інших мовах програмування, а повертає ці значення.

Якщо ми ділимо на нуль позитивне число, то отримуємо *Infinity*, а якщо негативне-то *-Infinity*.

Що буде, якщо спробувати перемножити, наприклад, число і рядок, ось так: 3 * '3'? В результаті буде отримане число 9. Це означає, що JavaScript автоматично здійснює перетворення типів при необхідності.

Однак є нюанс: якщо спробуємо скласти рядок і число, то JavaScript складе їх як рядки, а не як числа, ось так: '3' + 3 вийде рядок '33', а не число 6.

У випадку, наприклад, з множенням JavaScript розумів, що не можна перемножити рядки, тому рядки переводив в числа і перемножував їх. А випадок зі складанням можна трактувати двояко: складати як рядки або як числа (плюс-то використовується як для складання рядків, так і чисел).

Боротися з цим можна наступному способом: потрібно зробити неприпустиму для рядків операцію, наприклад, так: +'3' + 3 - поставимо плюс перед рядком і вона перетвориться до числа.

Другий варіант такий: можна сказати JavaScript, що потрібно явно перетворити рядок до числа. Це робиться за допомогою функції Number, ось так: Number('3') + 3. В результаті вийде 6, а не '33'.

До чисел можуть перетворюватися не тільки рядків, але і будь-які інші типи даних, наприклад true теж можна перетворити до числа таким чином: Number(true).

Можна перетворювати і до інших типів за допомогою функцій Boolean, String та інших подібних.

Мова програмування JavaScript часто використовується для створення інтерактивних веб-сторінок. Код JavaScript може бути вставлений безпосередньо в HTML-код сторінки, або можна створити окремий файл з розширенням .js і викликати його на кількох сторінках.

У JavaScript є важливе поняття - "об'єктна модель". Можна уявити, що існує кілька ящиків, в яких лежать матрьошки. Щоб додати записку в потрібну матрьошку, потрібно відкрити всі матрьошки, які охоплюють обрану програмістом матрьошку.

Порядок запуску скриптів

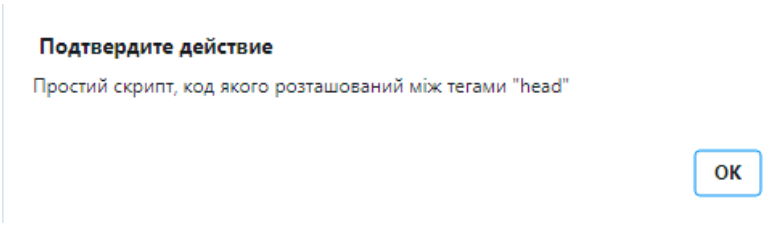
Розберемося, як запускається найпростіший скрипт на веб-сторінці. Для цього ми розглянемо код, який розташовується між тегами "head" на HTML-сторінці.

Лістинг програми і схема розташування коду скрипта на Web-сторінці:

```
<html>
 <head>
 …
 <script language="javascript">
 function func()
 {
alert(‘Простий скрипт, код якого розташований між тегами "head"’);
 }
 </script>
 …
 </head>
 <body>
 …
 <input type="Button" value="Клацнуть" onclick="func()">
 …
 </body>
 </html>
```

Зовнішнім проявом роботи скрипта стане зображення командної кнопки.

При клацанні по зображеній кнопці з'явиться повідомлення



Цей код містить функцію func(), яка виконується при натисканні на кнопку. Функція виводить повідомлення на екрані за допомогою методу alert().

Другий спосіб запуску скриптів полягає в тому, щоб створити окремий файл, який містить код скрипта.

Ось приклад коду, який можна зберегти в зовнішньому файлі з ім'ям script2.js:

```
function func2()
{
alert("Скрипт располагается во внешнем файле с именем script2.js");
}
```

Щоб підключити цей файл до HTML-коду, використовують атрибут src (джерело) тега script. Ось код, який потрібно вставити між тегами head:

```
<script Language="javascript" scr="script2.js"></script>
```

Третій спосіб запуску скриптів полягає в тому, щоб створити гіперпосилання з псевдо-URL.

Ось код, який запускає скрипт при натисканні на гіперпосилання:

```
<a href="javascript:alert('Скрипт запущений за допомогою гіперпосилання');"> Клацніть тут</a>
```

В результаті роботи скрипта на екрані з'явиться гіперпосилання:

Клацніть тут

При натисканні на гіперпосилання з'явиться вікно