

5. Основи роботи з функціями

Agenda

- Структура функції
- Види оголошення функцій
 - Function Declaration
 - Function Expression
 - Arrow Function
- IIFE
- Callback Function
- Рекурсія

Структура функції

CREATIVE
& TECH
PRJCTR
ONLINE
INSTITUTE


Оголошення функцій

Щоб створити функцію нам треба її *оголосити*.

```
function showMessage() {  
  console.log( 'Привіт!' );  
}
```

Змінні локальні та зовнішні

Локальні змінні



```
function showMessage() {  
  const message = 'Привіт!';  
  
  console.log(message);  
}
```

Змінні локальні та зовнішні

Зовнішні змінні

Функція може використовувати зовнішні змінні, наприклад:

```
const message = 'Привіт!';

function showMessage() {
  console.log(message);
}
```

Чим менше зовнішніх змінних тим краще

Параметри

Ми можемо передати в функцію довільні дані використовуючи параметри.

```
function showMessage(message) {  
    console.log(message);  
}
```

```
showMessage('Привіт');  
showMessage('Вітаю!');
```

```
function showMessage(name) {  
    console.log(`Привіт, ${name}`);  
}
```

```
showMessage('Анна');
```

Значення за замовчуванням

Якщо викликати функцію без аргументів, тоді відповідні значення стануть **undefined**.

```
function showMessage(name) {  
  console.log(`Вітаю, ${name}`);  
}  
  
showMessage(); // "Вітаю, undefined"
```

```
function showMessage(name = 'незнайомець') {  
  console.log(`Вітаю, ${name}`);  
}  
  
showMessage(); // "Вітаю, незнайомець"
```


Повернення значення, return

В якості результату, функція може повертати назад значення в код, який викликав цю функцію.

```
function sum(a, b) {  
  return a + b;  
}  
  
const result = sum(1, 2);  
  
result; // 3
```

Повернення значення, return

Директива **return** може бути в будь-якому місці функції. Коли виконання досягає цієї директиви, функція зупиняється, і в код, який викликав цю функцію, повертається значення

```
function calc(a, b) {  
  if (typeof a === 'number' && typeof b === 'number') {  
    return a + b;  
  } else {  
    return "Будь ласка, передайте число"  
  }  
}  
  
const result = calc(5, 5);  
const result2 = calc(5, 5);  
  
console.log(result); // 10  
console.log(result2); // "Будь ласка, передайте число"
```

Повернення значення, `return`

Якщо у функції немає `return` або має порожній `return` = `return undefined`

Найменування функції

Функції виконують дії. Тому в їхніх іменах зазвичай використовують дієслова. Ім'я повинне бути лаконічним, повинне якнайточніше описувати, що робить функція, щоб кожен хто читає код зміг зрозуміти, що саме робить функція.

Поширена практика розпочинати ім'я функції зі словесного префіксу, який описує дію. В команді має бути домовленість щодо значення префіксів.

Найменування функції

Наприклад, функції, які починаються з префіксу "**show**" зазвичай щось **показують**.

Функції, які починаються з ...

"**get...**" – повертають значення,

"**calc...**" – щось обчислюють,

"**create...**" – щось створюють,

"**check...**" – щось перевіряють і повертають булеве значення.

Q&A



CREATIVE
& TECH
CTR
PRJ
ONLINE
INSTITUTE

Види оголошення функцій

CREATIVE
& TECH
PRJ^{CT}TR
ONLINE
INSTITUTE

Function Declaration

Можна викликати до оголошення



```
sum( );
```

```
function sum(a, b) {  
  return a + b;  
};
```

CREATIVE
& TECH
PROJECT
ONLINE
INSTITUTE

Function Expression

Завантажуються лише тоді, коли інтерпретатор досягає цього рядка коду.



```
const sum = function(a, b) {  
  return a + b;  
};
```



```
sum(); // Буде помилка  
  
const sum = function(a, b) {  
  return a + b;  
};  
  
sum(); // Відпрацює коректно
```

Arrow function

Існує ще один простий та короткий синтаксис для створення функцій

```
const sum = (a, b) => {  
  return a + b  
};  
  
// Короткий запис  
const sum = (a, b) => a + b;  
  
// Запис з одним параметром  
const squared = a => a * a;  
  
// Без параметрів (дужки порожні)  
const sayHello = () => console.log('Hello!');
```

IIFE - Immediately Invoked Function Expression

Не рекомендується використовувати

```
(function () {  
  console.log('Привіт!')  
})();
```

Callback Function (або ЗВОРОТНИЙ ВИКЛИК)

Колбек-функція (або зворотний виклик) - це функція, передана в іншу функцію як аргумент, яка потім викликається після завершення будь-якої дії.

```
function greeting(name) {  
    alert(`Hello, ${name}`);  
};  
  
function processUserInput(callback) {  
    const name = prompt("Please enter your name.");  
    callback(name);  
};  
  
processUserInput(greeting);
```


Q&A



CREATIVE
& TECH
CTR
PRJ
ONLINE
INSTITUTE

Рекурсія

CREATIVE
& TECH
PRJ^{CT}TR
ONLINE
INSTITUTE

Рекурсія - це коли функція викликає сама себе.

Рекурсивний метод - це альтернатива ітеративному

pow - приведення до степеня

Ітеративний варіант - цикл for

```
function pow(x, n) {  
  let result = 1;  
  
  // множимо result на x n разів в циклі  
  for (let i = 0; i < n; i++) {  
    result *= x; // result = result * x(2)  
  }  
  
  return result;  
}  
  
console.log(pow(2, 3)); // 8
```

pow - приведення до степеня

Рекурсивний варіант

```
function pow(x, n) {  
  if (n === 1) {  
    return x;  
  }  
  
  return x * pow(x, n - 1);  
}  
  
console.log(pow(2, 3)); // 8
```

як це працює?

`pow(2, 3) = 2 * pow(2, 2)`

`pow(2, 2) = 2 * pow(2, 1)`

`pow(2, 1) = 2`

як це працює?

Контекст виконання – спеціальна внутрішня структура даних, яка містить інформацію про виклики функцій.

- Виконання поточної функції припиняється.
- Контекст виконання, пов'язаний з нею, запам'ятовується у спеціальній структурі даних – стеку контекстів виконання.
- Здійснюються вкладені виклики, для кожного з яких створюється контекст виконання.
- Після завершення старий контекст дістається зі стека, і виконання зовнішньої функції відновлюється з того місця, де вона була зупинена.

Q&A



CREATIVE
& TECH
CTR
PRJ
ONLINE
INSTITUTE

Про що варто
пам'ятати?

CREATIVE
& TECH
PROJECT
ONLINE
INSTITUTE

Використовуйте функцію для однієї задачі

Використовуйте зрозумілі найменування

Hoisting

- Hoisting (підйом) означає доступність функцій і змінних «у верхній частині» вашого коду, а не лише після їх створення. Об'єкти ініціалізуються під час компіляції та доступні будь-де у вашому файлі.
- Function declarations піднімаються, а function expressions – ні.

Коментарі

```
/**
 * Повертає x у n-му ступені.
 *
 * @param {число} x Число для підвищення.
 * @param {число} n Степінь має бути натуральним числом.
 * @return {число} x у n-му ступені.
 */

function pow(x, n) {
  // ...
}
```


Q&A



CREATIVE
& TECH
CTR
PRJ
ONLINE
INSTITUTE

Екстра матеріали

Про function declaration i function expression:

<https://www.freecodecamp.org/news/when-to-use-a-function-declarations-vs-a-function-expression-70f15152a0a0/>

Рекурсія:

<https://developer.mozilla.org/en-US/docs/Glossary/Recursion?retiredLocale=uk>

Hoisting:

<https://developer.mozilla.org/en-US/docs/Glossary/Hoisting>

Що далі?

Домашка

Сети та Мапи

CREATIVE
& TECH
PRJ^{CT}
ONLINE
INSTITUTE

Дякую
за увагу

Ocbita
& Future

CREATIVE
& TECH
CTR
PRJ
ONLINE
INSTITUTE