



# JavaScript Базовий

Замикання

# JavaScript Базовий

## Introduction



Охріменко Дмитро  
MCT

 \_okhrimenko

 dmitriy.okhrimenko

 dokhrimenko



MCID: 9210561

# JavaScript Базовий

Тема уроку

Замикання

# JavaScript Базовий

## План уроку

1. Глобальний об'єкт.
2. Контекст виконання (Execution Context).
3. Лексичне оточення (Lexical Environment).
4. Замикання (Closures).

# JavaScript Базовий

## Execution Context

**Контекст виконання (execution context)** – механізм, що описує оточення, в якому оцінюється та виконується JavaScript-код. Будь-який JavaScript-код працює всередині певного контексту виконання.

**Стек виконання (execution stack)** – зберігає всі контексти виконання, які були створені у процесі роботи коду.


Типи контекстів виконання:

- **Global** Execution Context – створюється один раз під час запуску сценарію.
- **Functional** Execution Context – створюється під час кожного запуску функції.
- **Eval Function** Execution Context – код, який виконується через eval.



# JavaScript Базовий

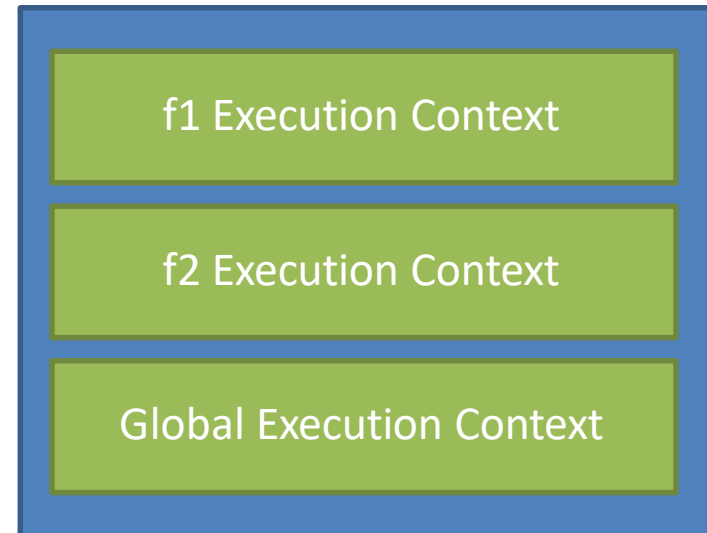
## Execution Stack



```
<script>
  function f1() {
    console.log("message");
  }

  function f2() {
    f1();
  }

  f2();
</script>
```



Execution Stack

# JavaScript Базовий

## Execution Context

Створення контексту виконання складається із двох фаз:

- **Creation Phase** – створюються та ініціалізуються компоненти контексту виконання.
- **Execution Phase** – виконання коду.

Execution context складається з:

- **LexicalEnvironment** – визначає зв'язок між ідентифікаторами та їхніми значеннями.
- **VariableEnvironment** – для ідентифікаторів, які створені через var.


LexicalEnvironment складається з:

- **EnvironmetRecord** – зберігає змінні та функції.
- **OuterEnv** – посилання на зовнішній LexicalEnvironment, якщо є.
- **ThisBinding** – значення this.

# JavaScript Базовий

## Lexical Environment

```
let y = 10;  
  
function sum() {  
  let x = 20;  
  console.log(x + y);  
}  
  
sum();
```



### Global Lexical Environment

ThisBinding: window  
OuterEnv: null  
EnvironmentRecord: {  
 y : 10,  
 sum: ref to function  
}

### Sum Lexical Environment

ThisBinding: window  
OuterEnv: Global Lexical Environment  
EnvironmentRecord: {  
 x : 20,  
}



# JavaScript Базовий

## Closure

**Замикання (closure)** – це функція та лексичне оточення, в якому ця функція була створена.

```
function makeCounter() {  
  let counter = 0;  
  
  function increment() {  
    return counter += 1;  
  }  
  
  return increment;  
}
```

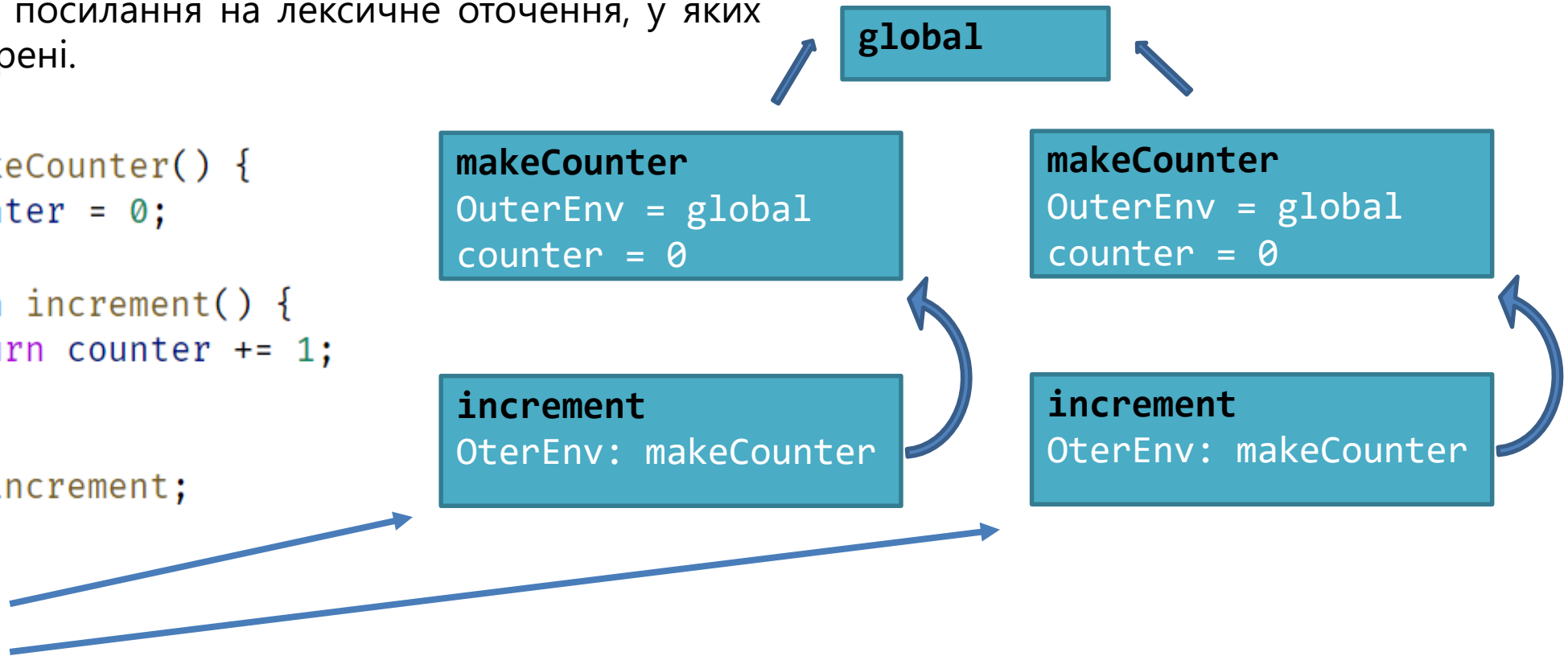


# JavaScript Базовий

## Closures

Функції містять посилання на лексичне оточення, у яких вони були створені.

```
function makeCounter() {  
  let counter = 0;  
  
  function increment() {  
    return counter += 1;  
  }  
  
  return increment;  
}
```



# JavaScript Базовий

## Висновки

- **globalThis** – стандартизоване ім'я глобального об'єкта.
- Варто уникати глобальних ідентифікаторів для зменшення ймовірності конфліктів з іншими сценаріями.
- **Execution Context** – механізм, що описує оточення, в якому виконується JavaScript-код.
- **Lexical Environment** – частина контексту виконання, яка використовується для роздільної здатності ідентифікаторів.
- **Замикання** – це функція та лексичне оточення, у якому ця функція була створена. Застосовується в різних шаблонах кодування на JavaScript.

# JavaScript Базовий

Дякую за увагу! До нових зустрічей!



Охріменко Дмитро  
МСТ



MCID: 9210561

# Інформаційний відеосервіс для розробників програмного забезпечення

