hillel IT school

Lesson 6

План заняття

- Декомпозиція та псевдокод;
- Види циклів;
- Цикл while;
- Цикл do while;
- Нескінченні цикли;
- Цикл for;
- Цикл for...in;
- Цикл for...of;
- break, continue;
- Вкладені цикли;

План заняття

- Декомпозиція та псевдокод;
- Види циклів;
- Цикл while;
- Цикл do while;
- Нескінченні цикли;
- Цикл for;
- Цикл for...in;
- Цикл for...of;
- break, continue;
- Вкладені цикли;

Цикли, вступ

При написанні скриптів часто постає завдання зробити однотипні дії багато разів.

Цикл – це спосіб повторити один і той же код кілька разів.

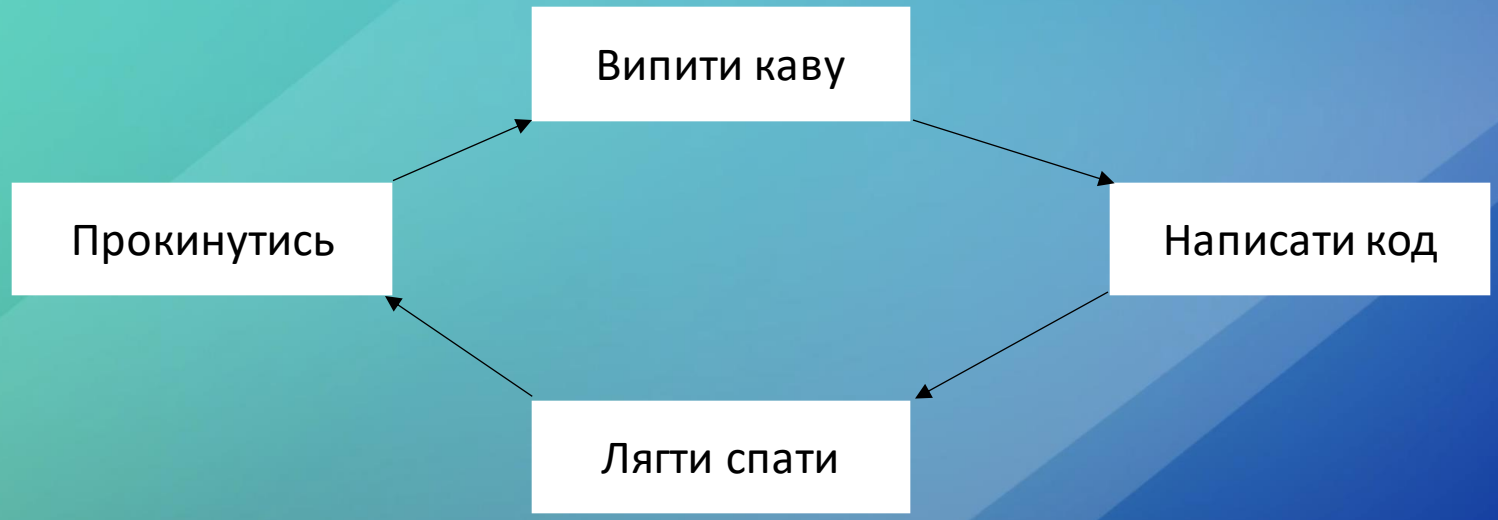
Одне виконання циклу називається ітерацією.

Доки умова є true, виконується код із тіла циклу. Якщо умова завжди true то цикл нескінченний.

Цикл блокує виконання коду доки він не завершиться!

Приклади у житті:

1. Робота;
2. Отримання готівки;
3. Хатні справи;
4. Пори року;
5. День та ніч;
6. Усе що повторюється...



Цикл while

Такий цикл називається циклом з передкмовою. Спочатку перевірили умову, потім виконали тіло циклу.

Будь-який вираз або змінна може бути умовою циклу. Умова виконується і конвертується у логічне значення.

Синтаксис:

```
while (умова) {
  // код
  // так зване "тіло циклу"
}
```

Якщо тіло цикла має тільки одну операцію, ми можемо опустити фігурні дужки {...}:

```
while (умова) операція;
```

Цикл do while

Такий цикл називається циклом з постумовою. Спочатку виконали тіло цикл, потім перевірили умову.

Синтаксис:

```
do {
  // тіло циклу
} while (умова);
```

Цю форму синтаксису слід використовувати лише тоді, коли ви хочете, щоб тіло циклу виконалось хоча б один раз, незалежно від умови.

Цикл for

Синтаксис:

```
for (початок; умова; крок) {
  // ... тіло циклу ...
}
```

Початок - виконується один раз, при вході в цикл.

Умова - перевіряється перед кожною ітерацією циклу. Якщо умова невірна, цикл зупиняєтья.

Тіло - виконується знову і знову, поки умова є правдивою (true).

Крок - виконується після тіла на кожній ітерації, але перед перевіркою умови.

Алгоритм:

- *Початок* виконання
- (Якщо *умова* == true → виконати тіло і виконати крок)
- ...

Цикл for

for (let i = 0; i < 3; i++) { … }

В цьому прикладі всередині циклу оголошена змінна і, яка виконує функцію лічильника. Це так зване «вбудоване» оголошення змінної. Такі змінні доступні лише всередині циклу.

Будь-яку частину for можна пропустити!

Нескінченний цикл

```
for (;;) {
  // буде вічно повторюватися
}
```

; повинні бути, інакше виникне синтаксична помилка.

Цикл for

hillel IT school

break – дериктива переривання циклу. чудова річ для тих ситуацій, коли умова для переривання знаходиться не на початку або кінці циклу, а всередині (або навіть в декількох місцях) тіла циклу;

contonue – пропуск ітерації. Вона не зупиняє весь цикл. Натомість, вона зупиняє поточну ітерацію і починає виконання циклу спочатку з наступної ітерації (якщо умова циклу досі вірна).

Її зручно використовувати коли закінчили з поточною ітерацією і хочемо продовжити з наступної.

Майте на увазі, що такі синтаксичні конструкції, які не є виразами, не можуть використовуватися з тернарним оператором ?. Власне, такі директиви як break/continue там не дозволені. Наприклад:

(i > 5) ? console.log(i) : continue;

Вложені цикли for

Ви можете вкладувти цикл у цикл стільки раз скільки вам потрібно. Будьте уважні це призведе до перенавантаження коду і зробить ваш код може блокувати програму.

Намагайтесь не робити вложеність циклу не більше 1 разу тобто for in for is good but for in for in for ….. is bad.

Цикл for мітки

Деколи нам потрібно вийти з кількох вкладених циклів. Мітка складається з ідентифікатора та двокрапки перед циклом. Ми не можемо використовувати мітки, щоб стрибати в довільне місце в коді.

Ситтксис:

```
outer: for (let i = 0; i < 3; i++) {
  for (let j = 0; j < 3; j++) {
    let input = prompt(`To do`, '');
    if (!input) break outer;
    // action
  }
}
```

```
outer:
for (let i = 0; i < 3; i++) { ... }
```

Цикл for ... of

Оператор for...of виконує цикл обходу об'єктів, що ітеруються (en-US) (включаючи Array, Map (en-US), Set, об'єкт аргументів (en-US) і подібних), викликаючи на кожному кроці ітерації оператори для кожного значення з різних властивостей об'єкта.

Синтаксис:

```
for (variable of iterable) {
  statement
}
```

Можна також використовувати const замість let, якщо не потрібно перепризначати змінні всередині блоку.

Для for...of обхід відбувається відповідно до того, який порядок визначений в об'єкті, що ітерується.

Цикл for ... in

Цикл for...in проходить тільки за властивостями, що перераховуються. Об'єкти, створені вбудованими конструкторами, такими як Array і Object мають властивості, що не перераховуються від Object.prototype і String.prototype, наприклад, від String-це indexOf(), а від Object - метод toString(). Цикл пройде за всіма властивостями об'єкта, що перелічуються, а також тим, що він успадкує від конструктора прототипу (властивості об'єкта в ланцюгу прототипу).

Синтаксис:

```
for (variable in object) {...
}
```

Дякую за увагу