УДК 004.4

INTRODUCTION IN ECMASCRIPT 6 FEATURES

Савчук Тамара, Куліковський Ігор

Вінницький національний технічний університет

Анотація

Представлено можливості нового стандарту ECMAScript 6 з метою кращого абстрагування, забезпечення стійкості структури додатку до зовнішніх та внутрішніх чинників.

Abstract

Presented abilities of the new standard ECMAScript 6 to better abstraction, ensuring the stability of the structure of the annex to external and internal factors.

Вступ

Розробка веб додатків (або single page application) - сайтів, робота яких повністю перенесена на сторону клієнта передбачає «спілкування» з сервером тільки «чистими» даними, без завантаження html-контенту. Кнопки, форми обробляються за допомогою javascript. Таким чином, сервер віддає тільки дані, як правило в форматі json, а сторона клієнта самостійно формує сторінку сайту, всі шаблони, списки, посилання, таблиці та інші оновлювані елементи.

Основні правила single page application:

3. Всі сутності веб-додатку засновані на моделях і об'єктах (всередині об'єктів інкапсульована робота з DOM-елементами сторінки).
4. HTML шаблони зберігаються в скриптах (наскільки це можливо).
5. Будь-які зміни на сторінці динамічно змінюють url.
6. Пряме завантаження будь-якого url повинна отримати приблизну сторінку з даними.
7. History back (кнопка назад в браузері) повинна оброблятися коректно і повертати сторінку в попередній стан.
8. Кешування моделей даних на стороні клієнта.

Концепція модулів як способу організації JavaScript-коду існувала давно. Коли додаток ускладнюється і код стає занадто потужним, виникає необхідність розбиття його на файли. В кожному файлі описується якась частина, що в подальшому повинні знову бути зібраними воєдино. Серед засобів, що використовують в таких випадках, слід відзначити такі як:

1. AMD - одна з найдавніших систем організації модулів, вимагає лише наявності клієнтської бібліотеки, наприклад, require.js, але підтримується і серверними засобами.
2. CommonJS - система модулів, вбудована в сервер Node.JS. Необхідна підтримка на клієнтський і серверній частині.
3. UMD - система модулів, яка запропонована в якості універсальної. UMD-модулі будуть працювати і в системі AMD і в CommonJS.

Всі перераховані вище системи вимагають використання різних бібліотек або систем збірки.

Модулі в стандарті ECMAScript (ES6) надають зручні засоби для виконання розбивки – збірки коду. Новий стандарт відрізняється від існуючих засобів перш за все тим, що це – стандарт, а тому, з часом, буде підтримуватися браузерами без додаткових утиліт.

```
function showMenu(title = "Без заголовку", width = 100, height = 200) {
  alert(`${title} ${width} ${height}`);
}
```

При передачі будь-якого значення, крім undefined, включаючи порожній рядок, нуль або null, параметр вважається переданим, і значення, за замовчуванням, не використовується.

В стандарті ES6 добавлені такі структури даних як Map та Set.

Map - це колекція записів виду ключ: значення, об'єкт та відрізняється тим, що перебирає завжди в порядку вставки і допускає будь-які ключі.

Основна область застосування Map - ситуації, коли рядкових ключів не вистачає та потрібно зберігати відповідності для ключів-об'єктів, або коли строковий ключ може бути абсолютно довільним.

Set - це колекція унікальних елементів, що також допускає будь-які ключі.

Наприклад, в звичайному об'єкті ОБ'ЄКТ можна використовувати абсолютно будь-які ключі. Є вбудовані методи, і є властивість з назвою __proto__, що зарезервована системою. Якщо назва ключа задається відвідувачем сайту, то він може спробувати використовувати таку властивість, замінити прототип, а це, при запуску JavaScript на сервері, вже може привести до серйозних помилок.

WeakMap і WeakSet — зменшені за функціоналом варіанти Map та Set, які дозволяють тільки звертатися до елементів за конкретномим ключем або значенням. Вони не перешкоджають збірнику сміття - лише якщо посилання на об'єкт залишилася тільки в WeakSet / WeakMap, воно буде видаленим.

У сучасному JavaScript з'явився новий, «привабливіший» синтаксис для класів:

```
class Назва [extends Батьківський клас]  {
  constructor
  методи
}
```

Функція constructor запускається при створенні new User, інші методи записуються в User.prototype. В обох випадках new User буде створювати об'єкти.

Метод sayHi також в обох випадках знаходиться в прототипі.

Але при оголошенні через class є ряд відмінностей:

- User можна викликати без new, буде помилка.
- Оголошення класу з точки зору області видимості поводиться як let. Зокрема, воно видно тільки в поточному блоці і тільки в коді, який знаходиться нижче оголошення (Function Declaration видно і до оголошення).

Генератори - це новий вид функцій в сучасному JavaScript. Вони відрізняються від звичайних тим, що можуть призупиняти своє виконання, повертати проміжний результат і далі відновлювати його пізніше, в довільний момент часу. Генератори створюються за допомогою функцій-генераторів

```
function * (...) { ... }.
```

Усередині генераторів і тільки всередині них дозволений оператор yield. Це іноді створює незручності, оскільки в коллбеках .map або .forEach виконати yield не можливо. Втім, можна виконати yield для масиву Зовнішній код і генератор обмінюються проміжними результатами за допомогою викликів next або yield. Генератори дозволяють писати плоский асинхронний код, за допомогою бібліотеки Babel.

Отже, використання стандарту ECMAScript 6 зростає відповідно до популярності web-додатків таких, як Single page, підтримується близько 80% браузерами та, за відсутністю конфліктів з ECMAScript 5, дає змогу використовувати вже розроблені модулі.

Список використаних джерел:

1. Дуглас Крокфорд. JavaScript сильні сторони. - Санкт-Петербург, Мир книг, 2013. 147 с.

2. Босуєл Д. Читаемый код или Програмирование как искуство. / Д. Босуєл, Т. Фаучел. - Мир книг, 2012. 152 с.

3. MOZILLA DEVELOPER NETWORK [Електронний ресурс] : [JavaScript reference] Режим доступу: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference.