Інструменти розробника браузера: Ваш путівник у світ DevTools

Інструменти розробника є у всіх найпопулярніших веб-браузерах. Вони надають приблизно однакову функціональність. Ми з вами будемо розглядати їх на прикладі браузера Google Chrome.

Chrome DevTools — це набір інструментів веб-розробника, вбудованих безпосередньо у браузер Google Chrome. DevTools може допомогти вам редагувати сторінки на льоту та швидко діагностувати проблеми, що зрештою допоможе вам покращити якість ваших веб застосунків.

Швидке відкриття

Command+Option+C (Mac) або Control+Shift+C (Windows, Linux, ChromeOS)

Контекстне меню

Клацніть правою кнопкою миші на сторінці та оберіть "Inspect"

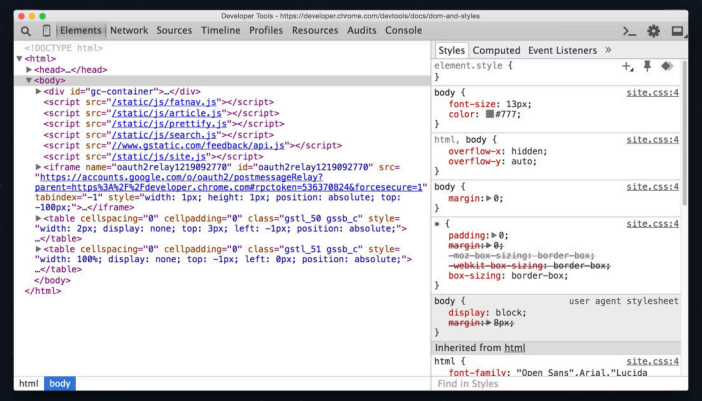
Багато вкладок

Інтерфейс DevTools має багато вкладок для різних завдань розробки

Вивчаємо DevTools: інтерфейс має багато вкладок, і ми розглянемо деякі з них, а саме ті, які використовуються найчастіше. Це потужний інструментарій, який стане вашим найкращим помічником у веб-розробці.

Elements: Вивчаємо структуру DOM

Тут ми можемо вивчати структуру DOM та діставати корисні атрибути з елементів, такі як class, id і т.п. Вкладка Elements — це ваш прямий доступ до HTML-структури сторінки в режимі реального часу.



Пошук елементів (нод)

Ви можете шукати елементи в дереві DOM за текстом, селектором CSS або селектором XPath. Це надзвичайно корисно при роботі зі складними веб-застосунками.

1. Наведіть курсор на панель "Елементи"
2. Натисніть Control+F або Command+F (Mac)
3. Введіть селектор або текст у рядок пошуку
4. Знайдений елемент буде виділено в дереві DOM

Редагування елементів

Ви можете відредагувати будь-який елемент в структурі DOM, змінивши контент, атрибути або взагалі замінивши назву тегу. Так ви можете відслідкувати, що буде відображено на сторінці в разі певних змін в DOM.

Просто клікніть 2 рази лівою кнопкою миші на елемент, який ви хочете відредагувати, і внесіть ваші зміни або відкрийте контекстне меню, клікнувши на потрібний елемент правою кнопкою миші.

Локальні зміни

Зміни, які ви внесете, доступні тільки вам — вони ніяк не впливають на те, що бачать інші користувачі. Як тільки ви перезавантажите сторінку або здійсните якусь навігацію, ваші зміни зникнуть. Це безпечне середовище для експериментів!

Screenshot елементу

Ви можете зробити скріншот будь-якого елементу на сторінці та зберегти зображення на свій пристрій. Це чудовий спосіб документувати компоненти або ділитися знахідками з командою.

Network: Аналізуємо мережеві запити

На цій вкладці ви можете відслідкувати, які запити були відправлені вашим застосунком. Кожен рядок мережевого журналу представляє ресурс. За замовчуванням ресурси перераховані в хронологічному порядку. Нижній ресурс — це те, що запитувалося останнім.

Розуміння стовпців мережевого журналу

01

Name

Назва ресурсу

02

Status

Код відповіді HTTP

03

Type

Тип ресурсу

04

Initiator

Що спричинило запит на ресурс

05

Size

Розмір ресурсу

06

Time

Тривалість запиту

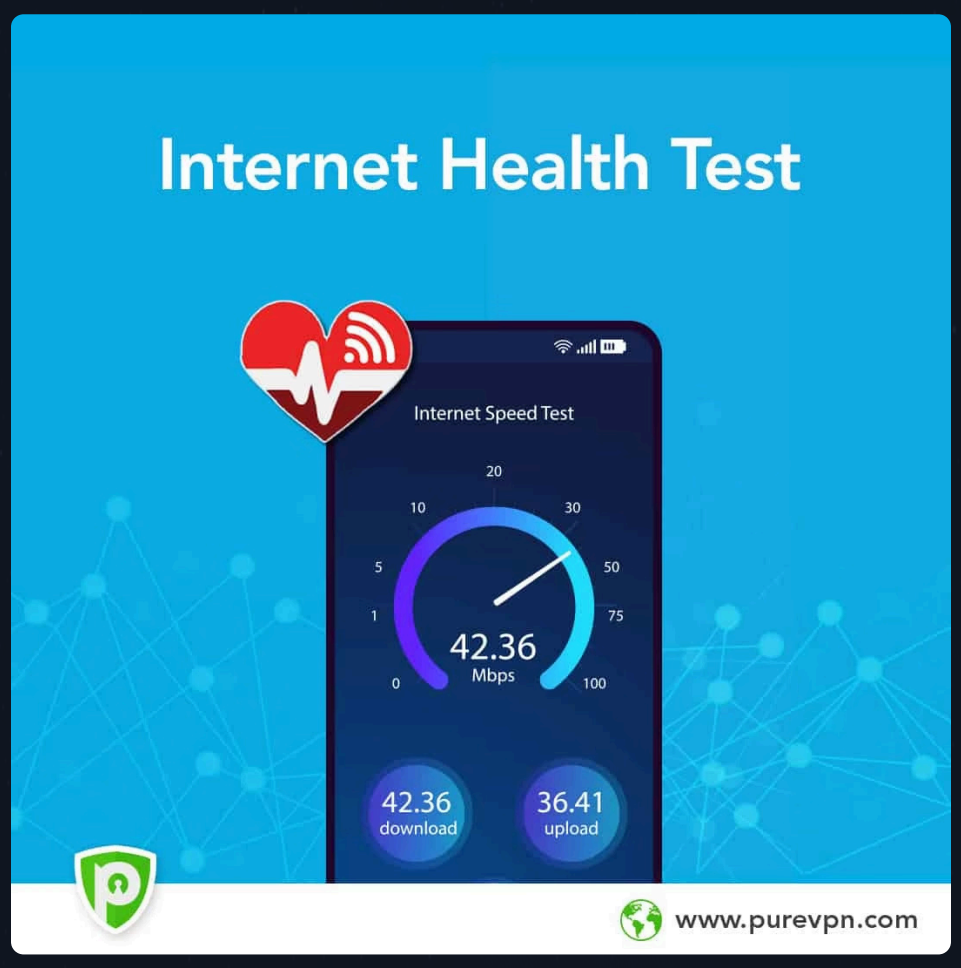
07

Waterfall

Графічне представлення етапів

Імітація повільного з'єднання

DevTools надає можливість імітувати різну швидкість з'єднання для того, щоб побачити, як користувачі будуть бачити ваш застосунок. Це дозволяє перевірити швидкість завантаження вашого сайту при повільному інтернеті, наприклад на мобільних пристроях.



Деталі запиту та відповідь

Коли ви зробите клік на певний запит в списку, у вас відкриється вікно з деталями цього запиту. Тут можна переглянути статус-код запиту, Request & Response headers та іншу корисну інформацію. У вкладці Payload можна переглянути тіло запиту. У вкладках Preview та Response буде міститися тіло відповіді, якщо воно є.

Application: Керування сховищем даних

На цій панелі є декілька розділів, але нас найбільше цікавить storage та його вміст. Давайте детально розглянемо різні типи веб-сховищ, які ви можете досліджувати та модифікувати за допомогою DevTools.

Local Storage

Об'єкти веб-сховища localStorage та sessionStorage дозволяють зберігати дані в браузері у вигляді пар ключ/значення.

Що цікаво, дані зберігаються навіть після оновлення сторінки (для sessionStorage) і після повного закриття і нового відкриття вікна браузера (для localStorage).

Ця вкладка надає вам можливість переглядати, редагувати, створювати та видаляти записи в localStorage.

Session Storage

Як було сказано раніше, особливістю Session Storage є те, що всі дані, які містяться у цьому сховищі, будуть очищені, коли ви закриєте вікно браузера.

В той час як дані в Local Storage є постійними, Session Storage ідеально підходить для тимчасових даних сесії користувача.

Ви можете отримати доступ до значень прямо з консолі. Це також стосується і Local Storage.

Основні методи localStorage

setItem()

Додає або оновлює значення у localStorage

```
localStorage.setItem('username', 'John');
```

getItem()

Отримує значення за вказаним ключем

```
const username =
localStorage.getItem('username');
```

removeItem()

Видаляє значення за вказаним ключем

```
localStorage.removeItem('username');
```

clear()

Очищує весь localStorage

```
localStorage.clear();
```

Приклад використання

```
// Зберігання значення у localStorage
localStorage.setItem('username', 'John');

// Отримання значення з localStorage
const username = localStorage.getItem('username');
console.log(username); // Виведе: John

// Видалення значення з localStorage
localStorage.removeItem('username');

// Очищення всього localStorage
localStorage.clear();
```

Cookies: Керування файлами cookie

Файли cookie HTTP в основному використовуються для керування сеансами користувачів, збереження налаштувань персоналізації користувача та відстеження поведінки користувачів. Розуміння структури та атрибутів cookies є критично важливим для веб-розробника.

Структура таблиці Cookies

- **Name** — Ім'я cookie
- **Value** — Значення cookie
- **Domain** — Хост з доступом
- **Path** — URL адреса для надсилання
- **Expires / Max-Age** — Термін життя
- **Size** — Розмір в байтах
- **HttpOnly** — Доступ з JavaScript
- **Secure** — Тільки HTTPS

Cookie Lifetime: Час життя cookies

З часом життя куків пов'язано кілька атрибутів, що встановлюються при їх створенні. Розуміння цих атрибутів допоможе вам ефективно керувати даними користувача.

Expires

Цей атрибут вказує дату та час, коли куки будуть видалені з пристрою користувача. Вказується у форматі GMT.

Session Cookies

Куки сесії не мають атрибута Expires або Max-Age. Вони існують лише протягом однієї сесії браузера.

1 2 3

Max-Age

Цей атрибут вказує час життя куки у секундах після її створення. Вона видаляється після вказаного часу.

Приклади встановлення атрибутів для часу життя куків

```
// Expires - куки будуть видалені 1 січня 2024 року
document.cookie = "expiresCookie=value; expires=Thu, 01 Jan 2024 00:00:00 GMT; path=/";

// Max-Age - куки будуть видалені через 1 годину
document.cookie = "maxAgeCookie=value; max-age=3600; path=/";

// Session Cookie - видаляться після закриття браузера
document.cookie = "sessionCookie=value; path=/";
```

Слід зазначити, що зазвичай cookie встановлюються сервером за допомогою надсилання заголовку Set-Cookie, де вказано інформацію про cookie. Ви можете модифікувати будь-які cookie, клікнувши на потрібний запис двічі лівою кнопкою миші, або видалити окремі чи всі cookies через контекстне меню.

Console: Потужний інструмент для JavaScript

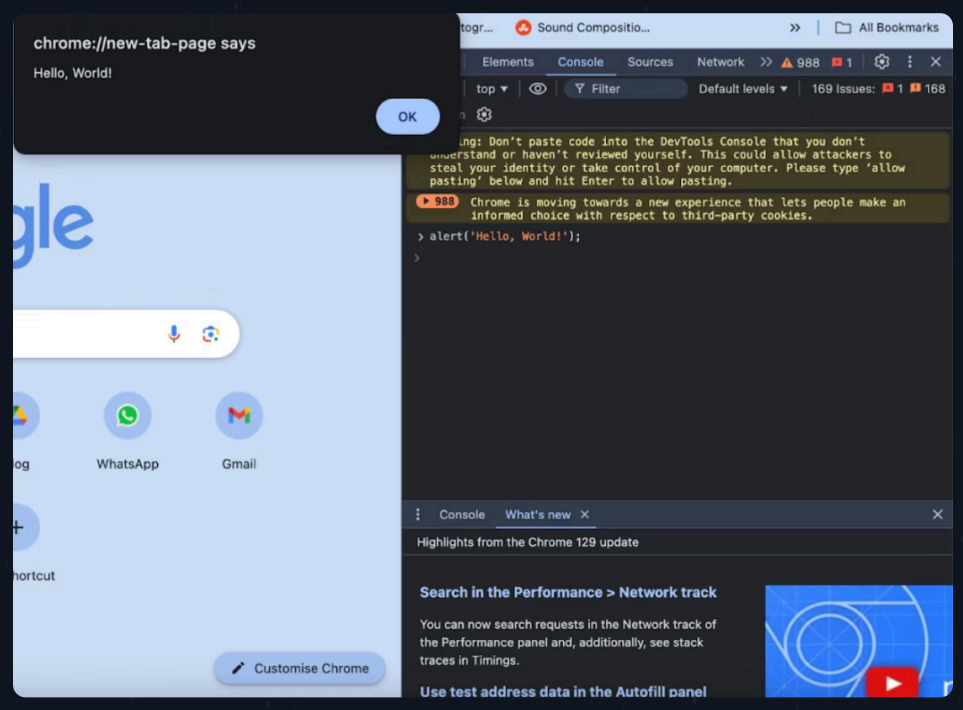
Вивід повідомлень в консоль

Веб-розробники часто записують повідомлення в консоль, щоб переконатися, що їхній JavaScript код працює належним чином. Щоб вивести повідомлення, ви вставляєте вираз на зразок `console.log('Hello, Console!')` у свій JavaScript.

Коли браузер виконує ваш JavaScript і бачить подібний вираз, він знає, що він має записати повідомлення в консоль. Це найпростіший та найефективніший спосіб дебагінгу вашого коду.

Виконання JavaScript коду

Ви також можете запускати будь-який JavaScript код в консолі і навіть мати доступ до DOM за допомогою JavaScript. Це перетворює консоль на потужну інтерактивну середу для експериментів та тестування.



Дебагінг коду

Використовуйте `console.log()`, `console.error()`, `console.warn()` та інші методи для виведення різних типів повідомлень. Це допоможе вам швидко знаходити та виправляти помилки.

Інтерактивне середовище

Консоль — це REPL (Read-Eval-Print Loop), де ви можете миттєво тестувати JavaScript код, викликати функції та перевіряти значення змінних без необхідності змінювати вихідний код.

Доступ до DOM

З консолі ви маєте повний доступ до Document Object Model. Використовуйте `document.querySelector()` та інші методи для маніпулювання елементами сторінки в реальному часі.

Приклад роботи з консоллю

```
<!DOCTYPE html>
<html>
<head>
  <title>Приклад консолі</title>
</head>
<body>
  <h1>Hello, World!</h1>
  <script>
    console.log('Сторінка завантажена!');
    const heading = document.querySelector('h1');
    console.log('Заголовок:', heading.textContent);
  </script>
</body>
</html>
```

Тепер ви володієте базовими знаннями про найважливіші вкладки Chrome DevTools! Продовжуйте практикуватися, експериментуйте з різними функціями, і ці інструменти стануть вашими найкращими помічниками у веб-розробці. Щасливого кодування! 🚀