Обробка винятків

№ уроку:	15	**Курс:**	JavaScript Базовий

Засоби навчання:	Visual Studio Code Web Browser

Огляд, мета та призначення уроку

Вивчити поняття винятків. Навчитися використовувати оператори try, catch і finally для обробки помилок етапу виконання.

Вивчивши матеріал цього заняття, учень зможе:

- Розуміти, що таке виняток.
- Розуміти різницю між помилкою етапу виконання та синтаксичною помилкою.
- Працювати з оператором try/catch.
- Використовувати оператор finally.
- Використовувати оператор throw для генерації користувацьких винятків.
- Створювати свої типи винятків.

Зміст уроку

1. Що таке виняток.
2. Конструкція try/catch.
3. Блок finally.
4. Використання throw.
5. Створення користувацьких винятків.

Резюме

- **Виняток** – ситуація, яка виникає через стан зовнішніх даних, пристроїв або інших систем, за якої подальше виконання програми немає сенсу.

- Наприклад, спроба десеріалізувати об'єкт з рядка JSON, який містить помилку, призведе до винятку. Оскільки функція JSON.parse має повернути об'єкт, але не може цього зробити через неправильні вхідні дані, вона викидає виняток (генерує помилку). Він зупиняє подальше виконання сценарію. Якщо parse не може повернути дані, у виконанні подальшого коду немає сенсу, тільки якщо розробник не передбачить альтернативного алгоритму й опрацює виняткову ситуацію.

- Якщо виняток генерується в коді, який розташований у блоці try, виняток припиняє виконання всіх подальших інструкцій в блоці try та переносить виконання в блок catch. Водночас інформація про помилку передається як параметр у блок catch. Після блоку catch сценарій продовжується. Якщо виняток виникає за межами блоку catch, припиняється виконання всього сценарію.

CyberBionic systematics

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine

Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

- **Об'єкт винятку містить три основні властивості:**
    1) **name** – назва винятку;
    2) **message** – опис помилки, яка виникла;
    3) **stack** – список викликів, які призвели до винятку.

- **Основні конструктори помилок:**
    - **Error** – помилка під час виконання;
    - **EvalError** – помилка під час виконання функції eval;
    - **RangeError** – значення виходить за певний діапазон;
    - **ReferenceError** – звернення до уявної змінної;
    - **SyntaxError** – синтаксична помилка під час виконання коду у функції eval;
    - **TypeError** – неприпустимий тип для змінної або параметра;
    - **URIError** – неприпустимі параметри для функції encodeURI та decodeURI.

- **Finally** – блок коду, який гарантовано виконується. Finally може йти відразу за блоком try або за блоком catch. Якщо під час виконання відбувається виняток у блоці try, спрацьовує блок catch. Після цього виконується блок finally. Якщо у блоці try не відбувся виняток, блок catch ігнорується, але finally однаково виконується. Якщо у блоці catch відбудеться ще один виняток або до блоку finally виконається оператор return, блок finally однаково виконається.

- Finally використовують для тих випадків, коли потрібно гарантувати виконання дії за будь-яких обставин. Наприклад, коли це операція, яка пов'язана зі звільненням ресурсів, виділених під час виконання блоку try.

- **Throw** – оператор, який використовується для створення винятку. Якщо цей оператор виконується в коді, створений виняток за стеком виклику передається до тих пір, поки не потрапить до блоку catch або поки не закінчаться адреси в стеку викликів. Цей оператор варто використовувати для бібліотечних функцій, які використовуються в різних частинах програми та мають повідомляти розробника про неправильне використання.

- Для **створення користувацького типу винятку** потрібно визначити новий клас і розширити один із системних класів винятків. За необхідністю визначити додаткові властивості та проініціалізувати властивості, які отримані у спадок від класу батька.

- Після ключового слова throw може використовуватися будь-який об'єкт або просте значення, яке буде винятком. Але браузер та інструменти для роботи з винятками можуть дати більше інформації та будуть зручними у використанні, якщо об'єкт помилки буде похідним від вбудованого класу помилки.

Закріплення матеріалу

- Що таке виняток? Наведіть приклад, коли у застосунку виникає виняток.
- Як працює блок try/catch?
- Назвіть основні властивості об'єкта винятку.
- Що таке finally? Опишіть принцип роботи цього блоку.
- Як створити користувацький тип винятку?

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine
Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Самостійна діяльність учня

Виконайте завдання у директорії Tasks\018 Try Catch. Текст завдань розташований у коментарях, у тегах script.

Рекомендовані ресурси

try/catch https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Statements/try...catch
Error та інші типи помилок
https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Error

CyberBionic systematics

CyberBionic Systematics ®
19 Evgenia Sverstiuka Str., 5 floor
Kyiv, Ukraine

Tel.:0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com