**JSON.parse() method**

**In JavaScript, the JSON.parse() method is used to parse JSON (JavaScript Object Notation) strings and convert them into JavaScript objects. JSON.parse() takes a JSON string as its parameter and returns a JavaScript object corresponding to the JSON data.**

**Syntax:**

JSON.parse(text [, reviver])

**text:**

**A JSON string to be parsed into a JavaScript object.**

**reviver (optional):**

**A function that can be used to transform the parsed value before it is returned. It's called for each key-value pair in the parsed JSON.**

**Example:**

*const* jsonString = '{"name": "John", "age": 30}';

*const* obj = JSON.parse(jsonString);

console.log(obj.name); // Output: John

console.log(obj.age); // Output: 30

**In this example:**

* **We have a JSON string jsonString representing a simple object with name and age properties.**
* **We use JSON.parse() to convert this JSON string into a JavaScript object obj.**
* **We can now access the properties of the JavaScript object obj using dot notation (obj.name, obj.age).**

**Using a Reviver Function:**

*const* jsonString = '{"name": "John", "age": 30}';

*const* obj = JSON.parse(jsonString, (*key*, *value*) *=>* {

    if (*key* === 'age') {

        return *value* + 10; // Add 10 to the age

    }

    return *value*;

});

console.log(obj.age); // Output: 40

**In this example:**

* **We use a reviver function to transform the parsed JSON value.**
* **In this reviver function, we check if the key is 'age'. If it is, we add 10 to the age value.**
* **As a result, the final value of the age property in the parsed object is 40.**

**Handling Invalid JSON:**

**If the JSON string passed to JSON.parse() is not valid JSON, a SyntaxError will be thrown.**

*const* invalidJsonString = '{"name": "John", "age": }'; // Invalid JSON

try {

*const* obj = JSON.parse(invalidJsonString);

} catch (error) {

    console.error('Invalid JSON:', error.message);

}

**In this example, invalidJsonString contains invalid JSON with a missing value for the age property. When we try to parse it using JSON.parse(), a SyntaxError is thrown, indicating that the JSON is invalid. We catch this error using a try-catch block and log the error message.**

**Example 1: Parsing JSON Array:**

// Example 1: Parsing JSON Array:

*const* jsonArrayString = '[1, 2, 3, 4, 5]';

*const* array = JSON.parse(jsonArrayString);

console.log(array); // Output: [1, 2, 3, 4, 5]

console.log(array.length); // Output: 5

console.log(array[0]); // Output: 1

**In this example, jsonArrayString contains a JSON array of numbers. We parse it using JSON.parse() and get back a JavaScript array.**

**Example 2: Parsing Nested JSON Objects:**

//Example 2: Parsing Nested JSON Objects:

*const* nestedJsonString = '{"person": {"name": "John", "age": 30}}';

*const* nestedObject = JSON.parse(nestedJsonString);

console.log(nestedObject.person.name); // Output: John

console.log(nestedObject.person.age); // Output: 30

**Here, nestedJsonString contains a JSON object with a nested object. We parse it using JSON.parse() and access properties of the nested object.**

**Example 3: Parsing JSON with Reviver Function:**

//Example 3: Parsing JSON with Reviver Function:

*const* jsonString = '{"name": "John", "age": 30}';

*const* parsedObject = JSON.parse(jsonString, (*key*, *value*) *=>* {

    if (*key* === 'age') {

        return *value* + 10;

    }

    return *value*;

});

console.log(parsedObject.age); // Output: 40

**In this example, we parse a JSON string and use a reviver function to modify the parsed value. Here, we add 10 to the age value.**

**Example 4: Handling Invalid JSON:**

//Example 4: Handling Invalid JSON:

*const* invalidJsonString = '{"name": "John", "age": }'; // Invalid JSON

try {

*const* obj = JSON.parse(invalidJsonString);

} catch (error) {

  console.error("Invalid JSON:", error.message);

}

**Here, invalidJsonString contains invalid JSON with a missing value for the age property. We try to parse it using JSON.parse() and catch the SyntaxError thrown due to invalid JSON.**

**Example 5: Parsing JSON Boolean Values:**

//Example 5: Parsing JSON Boolean Values:

*const* jsonString = '{"isStudent": true}';

*const* parsedObject = JSON.parse(jsonString);

console.log(parsedObject.isStudent); // Output: true

**In this example, we parse a JSON string containing a boolean value (true). After parsing, we get back a JavaScript object with the boolean property isStudent.**

**These examples demonstrate different use cases of JSON.parse() method, including parsing arrays, nested objects, using a reviver function, handling invalid JSON, and parsing boolean values.**