Ошибки в JavaScript и как их исправить

JavaScript может быть кошмаром при отладке: некоторые ошибки, которые он выдает, могут быть очень трудны для понимания с первого взгляда, и выдаваемые номера строк также не всегда полезны. Разве не было бы полезно иметь список, глядя на который, можно понять смысл ошибок и как исправить их? Вот он!  
  
Ниже представлен список странных ошибок в JavaScript. Разные браузеры могут выдавать разные сообщения об одинаковых ошибках, поэтому приведено несколько примеров там, где возможно.

Как читать ошибки?

Перед самим списком, давайте быстро взглянем на структуру сообщения об ошибке. Понимание структуры помогает понимать ошибки, и вы получите меньше проблем, если наткнетесь на ошибки, не представленные в этом списке.  
  
Типичная ошибка из Chrome выглядит так:

Uncaught TypeError: undefined is not a function

Структура ошибки следующая:

1. **Uncaught TypeError**: эта часть сообщения обычно не особо полезна. Uncaught значит, что ошибка не была перехвачена в catch, а TypeError — это название ошибки.
2. **undefined is not a function**: это та самая часть про ошибку. В случае с сообщениями об ошибках, читать их нужно прямо буквально. Например, в этом случае, она значит то, что код попытался использовать значение undefined как функцию.

Другие webkit-браузеры, такие как Safari, выдают ошибки примерно в таком же формате, как и Chrome. Ошибки из Firefox похожи, но не всегда включают в себя первую часть, и последние версии Internet Explorer также выдают более простые ошибки, но в этом случае проще — не всегда значит лучше.  
  
Теперь к самим ошибкам.

Uncaught TypeError: undefined is not a function

**Связанные ошибки:** number is not a function, object is not a function, string is not a function, Unhandled Error: ‘foo’ is not a function, Function Expected  
  
Возникает при попытке вызова значения как функции, когда значение функцией не является. Например:

var foo = undefined;

foo();

Эта ошибка обычно возникает, если вы пытаетесь вызвать функцию для объекта, но опечатались в названии.

var x = document.getElementByID('foo');

Несуществующие свойства объекта по-умолчанию имеют значение undefined, что приводит к этой ошибке.  
  
Другие вариации, такие как “number is not a function” возникают при попытке вызвать число, как будто оно является функцией.  
  
**Как исправить ошибку:** убедитесь в корректности имени функции. Для этой ошибки, номер строки обычно указывает в правильное место.

Uncaught ReferenceError: Invalid left-hand side in assignment

**Связанные ошибки:** Uncaught exception: ReferenceError: Cannot assign to ‘functionCall()’, Uncaught exception: ReferenceError: Cannot assign to ‘this’  
  
Вызвано попыткой присвоить значение тому, чему невозможно присвоить значение.  
  
Наиболее частый пример этой ошибки — это условие в if:

if(doSomething() = 'somevalue')

В этом примере программист случайно использовал один знак равенства вместо двух. Выражение “left-hand side in assignment” относится к левой части знака равенства, а, как можно видеть в данном примере, левая часть содержит что-то, чему нельзя присвоить значение, что и приводит к ошибке.  
  
**Как исправить ошибку:** убедитесь, что вы не пытаетесь присвоить значение результату функции или ключевому слову this.

Uncaught TypeError: Converting circular structure to JSON

**Связанные ошибки:** Uncaught exception: TypeError: JSON.stringify: Not an acyclic Object, TypeError: cyclic object value, Circular reference in value argument not supported  
  
Всегда вызвано циклической ссылкой в объекте, которая потом передается в JSON.stringify.

var a = { };

var b = { a: a };

a.b = b;

JSON.stringify(a);

Так как a и b в примере выше имеют ссылки друг на друга, результирующий объект не может быть приведен к JSON.  
  
**Как исправить ошибку:** удалите циклические ссылки, как в примере выше, из всех объектов, которые вы хотите сконвертировать в JSON.

Unexpected token ;

**Связанные ошибки:** Expected ), missing ) after argument list  
  
Интерпретатор JavaScript что-то ожидал, но не обнаружил там этого. Обычно вызвано пропущенными фигурными, круглыми или квадратными скобками.  
  
Токен в данной ошибке может быть разным — может быть написано “Unexpected token ]”, “Expected {” или что-то еще.  
  
**Как исправить ошибку:** иногда номер строки не указывает на правильное местоположение, что затрудняет исправление ошибки.  
  
Ошибка с [ ] { } ( ) обычно вызвано несовпадающей парой. Проверьте, все ли ваши скобки имеют закрывающую пару. В этом случае, номер строки обычно указывает на что-то другое, а не на проблемный символ.  
  
Unexpected / связано с регулярными выражениями. Номер строки для данного случая обычно правильный.  
  
Unexpected; обычно вызвано символом; внутри литерала объекта или массива, или списка аргументов вызова функции. Номер строки обычно также будет верным для данного случая.

Uncaught SyntaxError: Unexpected token ILLEGAL

**Связанные ошибки:** Unterminated String Literal, Invalid Line Terminator  
  
В строковом литерале пропущена закрывающая кавычка.  
  
**Как исправить ошибку:** убедитесь, что все строки имеют правильные закрывающие кавычки.

Uncaught TypeError: Cannot read property ‘foo’ of null, Uncaught TypeError: Cannot read property ‘foo’ of undefined

**Связанные ошибки:** TypeError: someVal is null, Unable to get property ‘foo’ of undefined or null reference  
  
Попытка прочитать null или undefined так, как будто это объект. Например:

var someVal = null;

console.log(someVal.foo);

**Как исправить ошибку:** обычно вызвано опечатками. Проверьте, все ли переменные, использованные рядом со строкой, указывающей на ошибку, правильно названы.

Uncaught TypeError: Cannot set property ‘foo’ of null, Uncaught TypeError: Cannot set property ‘foo’ of undefined

**Связанные ошибки:** TypeError: someVal is undefined, Unable to set property ‘foo’ of undefined or null reference  
  
Попытка записать null или undefined так, как будто это объект. Например:

var someVal = null;

someVal.foo = 1;

**Как исправить ошибку:** это тоже обычно вызвано ошибками. Проверьте имена переменных рядом со строкой, указывающей на ошибку.

Uncaught RangeError: Maximum call stack size exceeded

**Связанные ошибки:** Uncaught exception: RangeError: Maximum recursion depth exceeded, too much recursion, Stack overflow  
  
Обычно вызвано неправильно программной логикой, что приводит к бесконечному вызову рекурсивной функции.  
  
**Как исправить ошибку:** проверьте рекурсивные функции на ошибки, которые могут вынудить их делать рекурсивные вызовы вечно.

Uncaught URIError: URI malformed

**Связанные ошибки:** URIError: malformed URI sequence  
  
Вызвано некорректным вызовом decodeURIComponent.  
  
**Как исправить ошибку:** убедитесь, что вызовы decodeURIComponent на строке ошибки получают корректные входные данные.

XMLHttpRequest cannot load [some/url](http://some/url/). No ‘Access-Control-Allow-Origin’ header is present on the requested resource

**Связанные ошибки:** Cross-Origin Request Blocked: The Same Origin Policy disallows reading the remote resource at [some/url](http://some/url/)  
  
Эта проблема всегда связана с использованием XMLHttpRequest.  
  
**Как исправить ошибку:** убедитесь в корректности запрашиваемого URL и в том, что он удовлетворяет [same-origin policy](https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy). Хороший способ найти проблемный код — посмотреть на URL в сообщении ошибки и найти его в своём коде.

InvalidStateError: An attempt was made to use an object that is not, or is no longer, usable

**Связанные ошибки:** InvalidStateError, DOMException code 11  
  
Означает то, что код вызвал функцию, которую нельзя было вызывать в текущем состоянии. Обычно связано c XMLHttpRequest при попытке вызвать на нём функции до его готовности.

var xhr = new XMLHttpRequest();

xhr.setRequestHeader('Some-Header', 'val');

В данном случае вы получите ошибку потому, что функция setRequestHeader может быть вызвана только после вызова xhr.open.  
  
**Как исправить ошибку:** посмотрите на код в строке, указывающей на ошибку, и убедитесь, что он вызывается в правильный момент или добавляет нужные вызовы до этого (как с xhr.open).

Заключение

JavaScript содержит в себе одни из самых бесполезных ошибок, которые я когда-либо видел, за исключением печально известнойExpected T\_PAAMAYIM\_NEKUDOTAYIM в PHP. Большая ознакомленность с ошибками привносит больше ясности. Современные браузеры тоже помогают, так как больше не выдают абсолютно бесполезные ошибки, как это было раньше.