1. Try catch

Обычно скрипт в случае ошибки «падает» (сразу же останавливается), с выводом ошибки в консоль. Но есть синтаксическая конструкция try..catch, которая позволяет «ловить» ошибки и вместо падения делать что-то более осмысленное.

**Синтаксис «try…catch»**

Конструкция try..catch состоит из двух основных блоков: try, и затем catch:

try {

// код...

} catch (err) {

// обработка ошибки

}

**Работает она так:**

1. Сначала выполняется код внутри блока try {...}.
2. Если в нём нет ошибок, то блок catch(err) игнорируется: выполнение доходит до конца try и потом далее, полностью пропуская catch.
3. Если же в нём возникает ошибка, то выполнение try прерывается, и поток управления переходит в начало catch(err). Переменная err (можно использовать любое имя) содержит объект ошибки с подробной информацией о произошедшем.

Таким образом, при ошибке в блоке try {…} скрипт не «падает», и мы получаем возможность обработать ошибку внутри catch.

!!!!!!Чтобы try..catch работал, код должен быть выполнимым. Другими словами, это должен быть корректный JavaScript-код (try..catch  не будет работать если есть синтаксические ошибки ).

**Объект ошибки**

Когда возникает ошибка, JavaScript генерирует объект, содержащий её детали. Затем этот объект передаётся как аргумент в блок catch:

Для всех встроенных ошибок этот объект имеет два основных свойства:

**name**

Имя ошибки. Например, для неопределённой переменной это "ReferenceError".

**message**

Текстовое сообщение о деталях ошибки.

В большинстве окружений доступны и другие, нестандартные свойства. Одно из самых широко используемых и поддерживаемых – это:

**stack**

Текущий стек вызова: строка, содержащая информацию о последовательности вложенных вызовов, которые привели к ошибке. Используется в целях отладки.

Например:

try {

lalala; // ошибка, переменная не определена!

} catch(err) {

alert(err.name); // ReferenceError

alert(err.message); // lalala is not defined

alert(err.stack); // ReferenceError: lalala is not defined at (...стек вызовов)

// Можем также просто вывести ошибку целиком

// Ошибка приводится к строке вида "name: message"

alert(err); // ReferenceError: lalala is not defined

}

Однако возможна ситуация, что в блоке catch тоже возникнет ошибка, и тогда выполнение следующей строчки уже не случится.

В конструкцию try...catch можно добавить блок finally, который выполнится после блоков try и catch. Неважно какой код выполнился в предыдущих блоках, после их завершения (даже если из catch была выброшена новая ошибка) исполнится код в блоке finally.