**Определение**

1. Ключевое слово this — это специальная переменная, которая содержит в себе объект (контекст) выполняемого кода.

проще всего представить, что this — это уникальная переменная, которая хранит в себе контекст исполняемого кода.

И наоборот — контекст — это значение ключевого слова this.

**Что такое контекст?**

Контекст (контекст выполнения) (execution context) — это всегда значение ключевого слова this, которое является ссылкой на объект, который запустил метод (функцию).

**Что такое стек выполнения?**

Стек выполнения (execution stack), его ещё называют стеком вызовов (call stack), это LIFO-стек(last in, first out, «последним пришёл — первым ушёл»), который используется для хранения контекстов выполнения, создаваемых в ходе работы кода.

Интерпретатор JavaScript всегда начинает выполнять код с глобального контекста (в браузере это объект window).

Когда JS-движок начинает обрабатывать скрипт, движок создаёт глобальный контекст выполнения и помещает его в текущий стек. При обнаружении команды вызова функции движок создаёт новый контекст выполнения для этой функции и помещает его в верхнюю часть стека.

Движок выполняет функцию, контекст выполнения которой находится в верхней части стека. Когда работа функции завершается, её контекст извлекается из стека и управление передаётся тому контексту, который находится в предыдущем элементе стека.

После того, как весь код оказывается выполненным, движок извлекает глобальный контекст выполнения из текущего стека.

**От чего зависит значение this?**

От того, какая и когда функция запускается.

**Когда определяется значение this?**

В момент вызова функции. Только вызов функции – любой, кроме стрелочной, изменяет this. Cтрелочная функция это исключение – она никак не изменяет this.

**Что такое скрипт, а что такое модуль?**

Это разные способы выполнения кода. Скрипт — это тот код, который запускается по умолчанию. Модуль - запуск того же самого кода, с определенным параметром, который называется «модуль». Например, когда используется импорт.

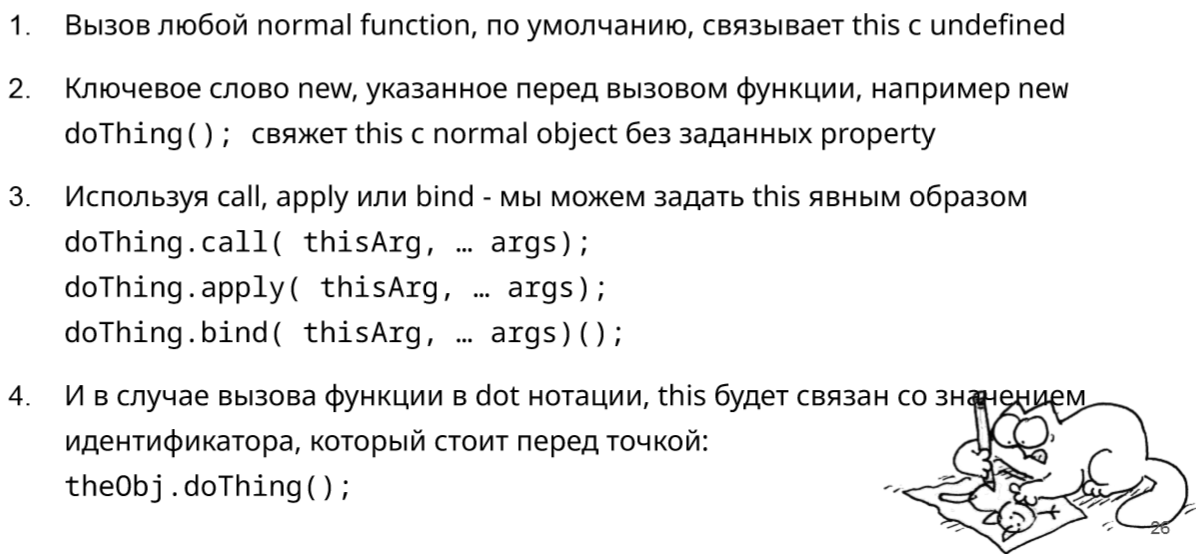
**Что такое нормальная функция?**

Функция, которая не стрелочная.

**Чем может быть this?**

Любым типом данных.

**Какие формы вызова меняют this?**

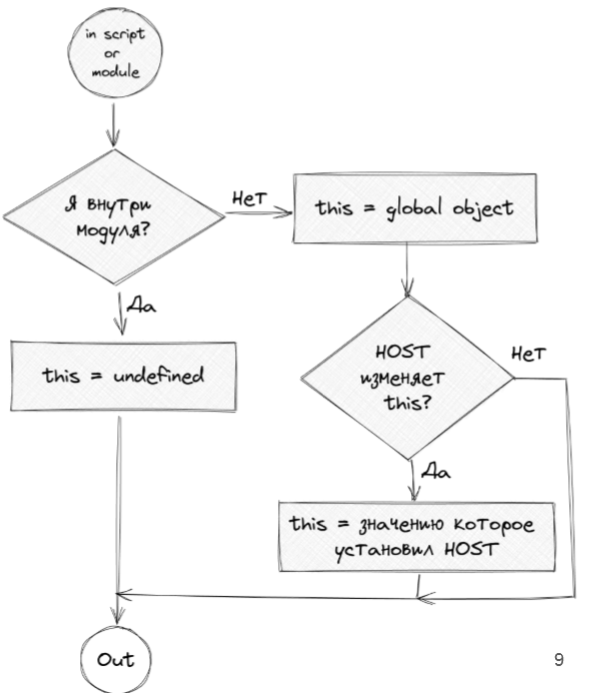


**С чем связан this в глобальном окружении (global environment)?**

Внутри модуля всегда только undefined.

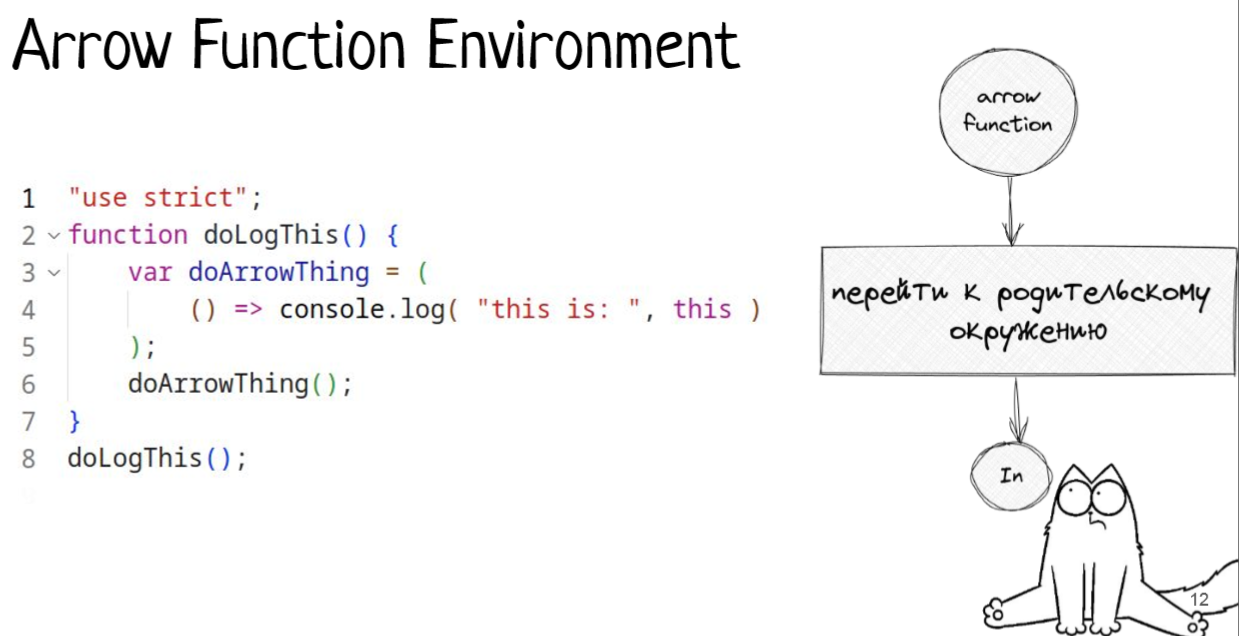
Если мы внутри скрипта, то this по умолчанию, связан с глобальным объектом. Но, согласно спецификации JS, HOST-среда (браузер, нода) имеет возможность изменять this. Например, в ноде this = {}.

Поэтому тут this будет зависеть от того меняет HOST-среда this или нет.



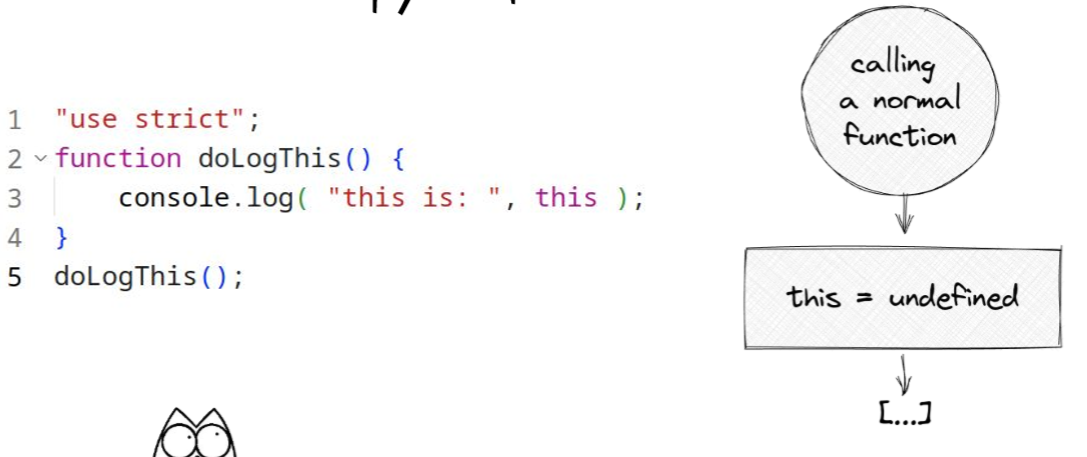
**Чему равно this в стрелочной функции?**

Если мы в стрелочной функции, то нужно перейти к родительском окружению, чтобы понять чему равно this.



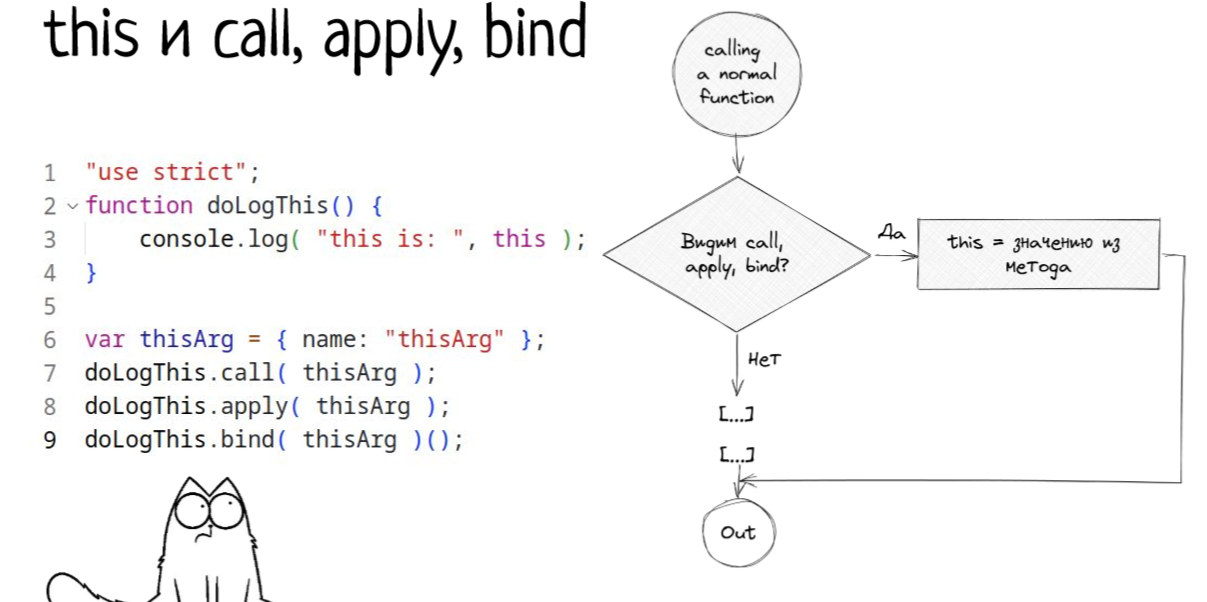
**Чему равно this в нормальной функции?**

Если this явно никак не задавался, то, по умолчанию он связан со значением undefined.



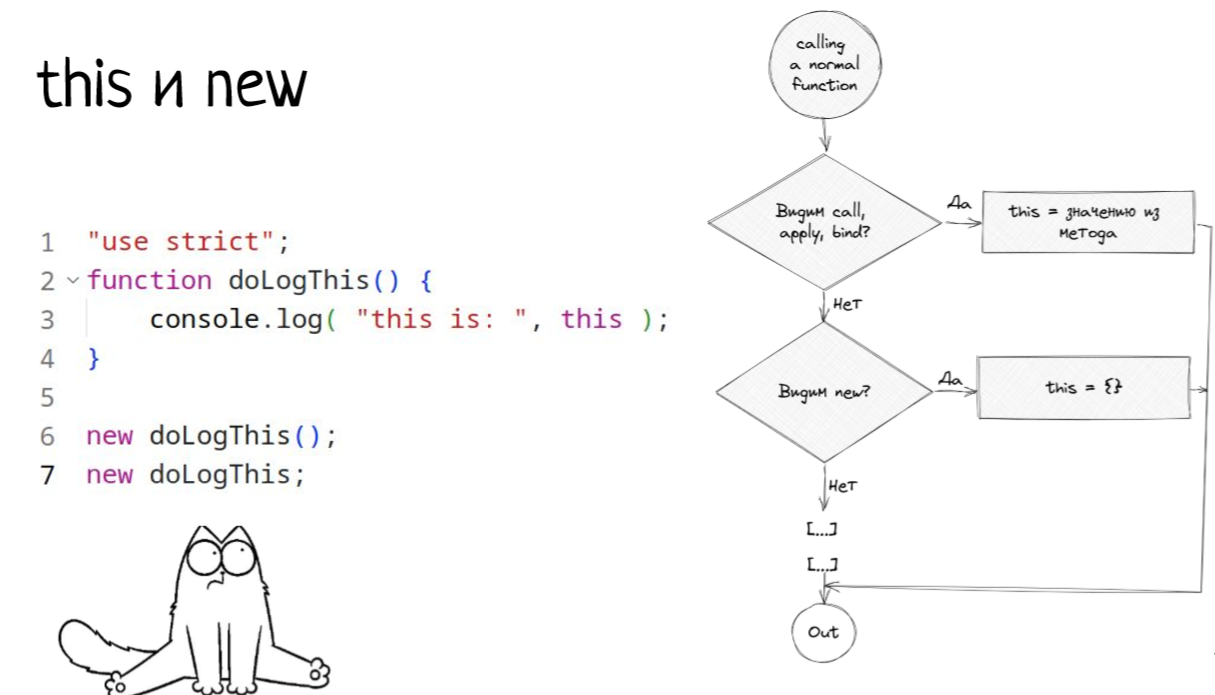
**Чему равно this если функция вызвана методами call, apply, bind?**

Эти методы явно задают this, поэтому он будет равен тому, с чем его явно связали.



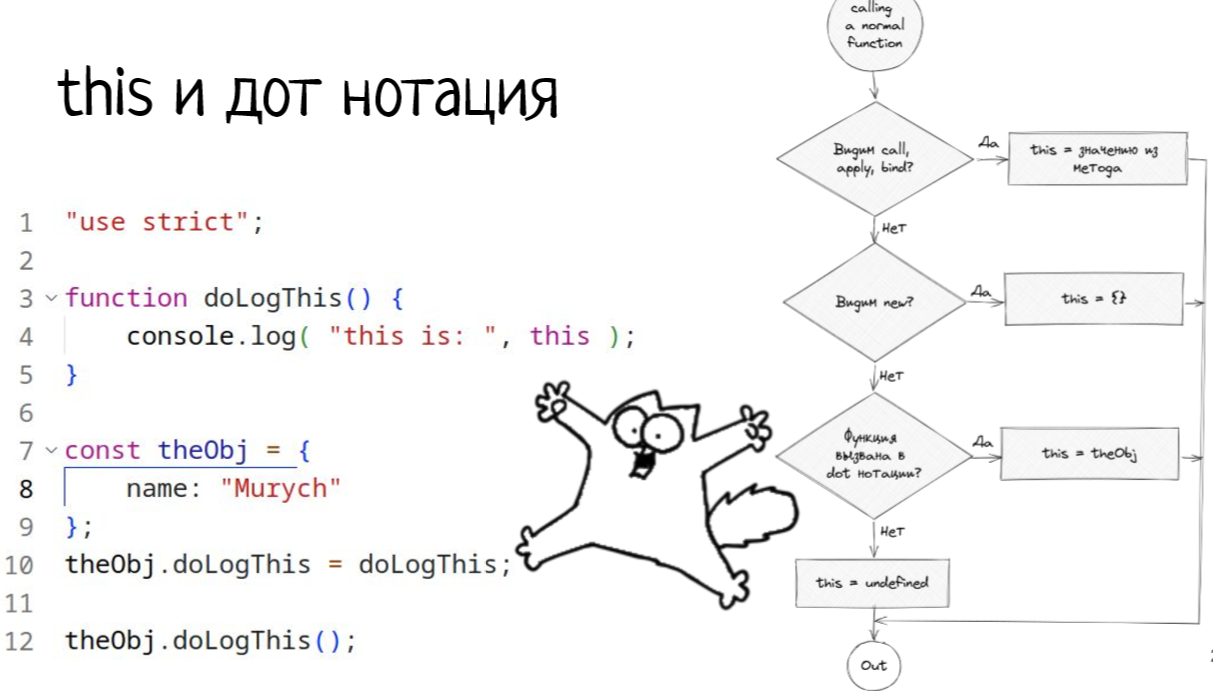
**Чему равно this если есть new?**

При вызове функции с new, this этой функции всегда связывается с пустым объектом.



**Чему равно this в дот нотации?**

This будет связан с идентификатором перед точкой.



**Задача 1.**

**Чему равно this ?**

"use strict";

const theObj = {

    name: "Murych",

    doLogThis: function () {

        console.log( "this is: ", this );

    }

};

theObj.doLogThis();

var doLogThisGlobal = theObj.doLogThis;

doLogThisGlobal();

**Задача 2.**

**Чему равно this?**

"use strict";

const theObj = {

    name: "Murych",

    doLogThis: function () {

        console.log( "this is: ", this );

    }

};

setTimeout( theObj.doLogThis, 1 );

**Задача 3.**

**Чему равно this?**

"use strict";

const theObj = {

    name: "Murych",

    doLogThis: function () {

        console.log( "this is: ", this );

    }

};

var doLogThis = theObj.doLogThis;

setTimeout( doLogThis, 1 );

**Задача 4.**

**Чему равно this?**

"use strict";

function doHandleClick() {

    console.log( "this is:", this )

}

document

    .body

    .addEventListener( "click", doHandleClick );

**Задача 5.**

**Чему равно this?**

"use strict";

function doHandleClick() {

    console.log( "this is:", this )

}

document

    .body

    .addEventListener( "click", doHandleClick.bind({yo: "yo"}) );

**Задача 6.**

**Чему равно this?**

"use strict";

function doLogThis() {

    var doArrowThing = (

        () =>

            console.log( "this is: ", this )

    );

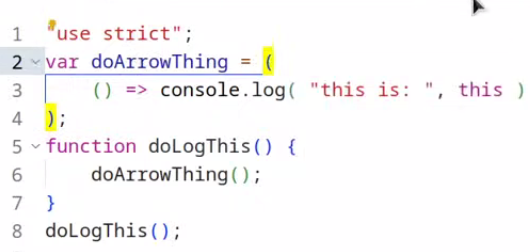
    doArrowThing();

}

doLogThis();

**Задача 7.**

**Чему равно this?**



**Задача 8.**

**Чему равно this?**

"use strict";

var doArrowThing = (

    () => console.log( "this is: ", this )

);

const theObj = {

    name: 'name',

    doSayYourName: function(){

        doArrowThing()

    }

}

theObj.doSayYourName()

**Задача 9.**

**Чему равно this?**

"use strict";

const theObj = {

    name: 'name',

    doSayYourName: function(){

        var doArrowThing = (

            () => console.log( "this is: ", this )

        );

        doArrowThing();

    }

}

theObj.doSayYourName()

**Задача 10.**

**Чему равно this?**

"use strict";

const theObj = {

    name: "theObj murych",

    returnFunction: function () {

        var doArrowThing = (

            () => (console.log( "this is: ", this ))

        );

        return doArrowThing;

    }

}

const theSuperObj = {

    name: "murych",

    doSayYourName: function ( doThing ) {

        doThing();

    }

}

theSuperObj.doSayYourName(theObj.returnFunction());

**Ответы.**

**1. Undefined.**

**2. Undefined.**

Видим call, apply, bind – нет. Видим new – нет. Функция вызвана в дот нотации – НЕТ. Тут через дот нотацию идет обращение к свойству, но вызова нет. Но в ноде, например там будет объект timeout, т.к. settimeout – т.к. нода это HOST-среда и она может изменять this.

**3. Undefined.**

Пример абсолютно идентичен предыдущему.

**4. Body.**

По всем правилам должно быть равно undefined, но т.к. addEventListener – это сторонний апи, он устанавливает this по своим правилам. По умолчанию это будет элемент перед точкой – т.е. body.

**5.** {yo: "yo"} – сработает bind, т.к. значение по умолчанию апи перебилось явной привязкой.

1/19

**6. Undefined.**

**7. Global object.** Т.к. мы находимся в стрелочной функции, необходимо перейти к ее родительскому окружению, не там, где она вызывается, а именно там, где this определяется, и это будет глобальное окружение.

**8. Global object.** В дот нотации вызвана не та функция, которая содержит this.

**9. theObj.** Т.к. this находится в стрелочной функции, необходимо обратиться к родительскому окружению. Это будет функция doSayYourName. Чтобы понять, чему равно this в этой функции, нужно посмотреть, как эта функция была вызвана. А вызвана она была через точку, значит this будет равен тому, что перед точкой, т.е. theObj.

**10. theObj.** Т.к. this находится в стрелочной функции, необходимо обратиться к родительскому окружению. Это будет returnFunction. Чтобы понять, чему равно this в этой функции, нужно посмотреть, как эта функция была вызвана. А вызвана она была через точку, от theObj.