

Node js

# Зачем???

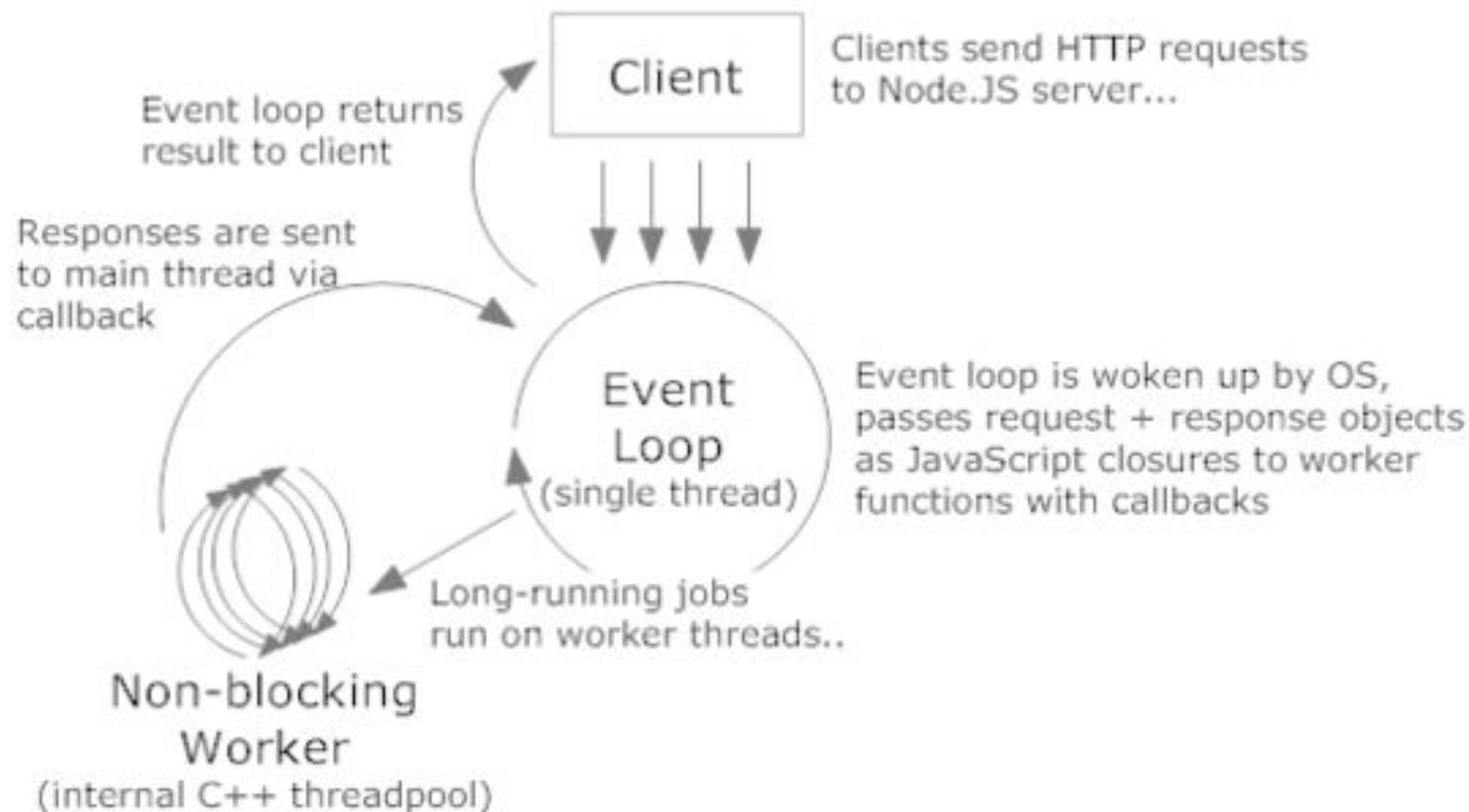
JavaScript живёт двумя, может даже тремя разными жизнями: весёлый маленький DHTML-помощник из середины 90-х годов, более серьёзный frontend-инструмент в лице jQuery и наконец серверный (server-side, backend) JavaScript.

Чтобы ваш JavaScript код выполнялся на *вычислительной машине вне браузера* (на **backend**), он должен быть интерпретирован и, конечно же, выполнен. Именно это и делает Node.js. Для этого он использует движок V8 VM от Google — ту же самую среду исполнения для JavaScript, которую использует браузер Google Chrome.

Таким образом, Node.js состоит из 2 вещей: среды исполнения и полезных библиотек.

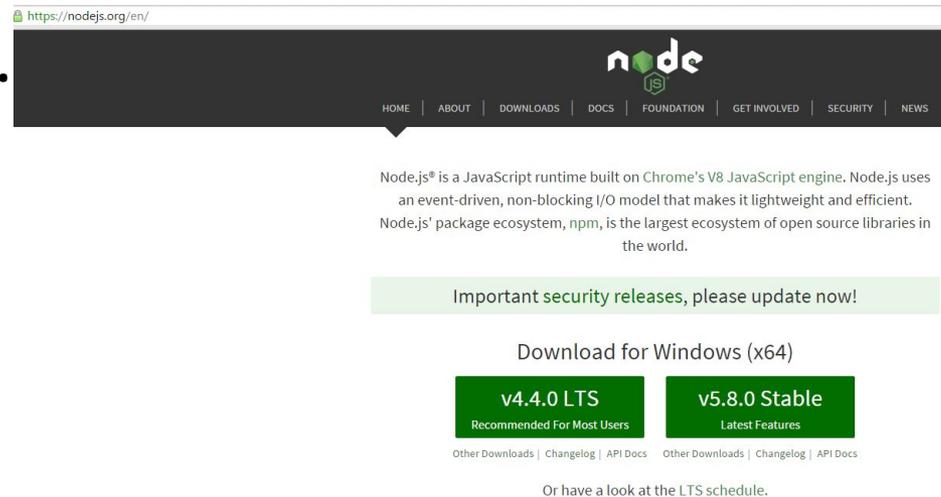
- **Node** или **Node.js** — программная платформа, основанная на движке **V8** (транслирующем **JavaScript** в машинный код), превращающая **JavaScript** из узкоспециализированного языка в язык общего назначения.
- **Node.js** добавляет возможность **JavaScript** взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из **JavaScript**-кода. **Node.js** применяется преимущественно на сервере. В основе **Node.js** лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

## Node.JS Processing Model



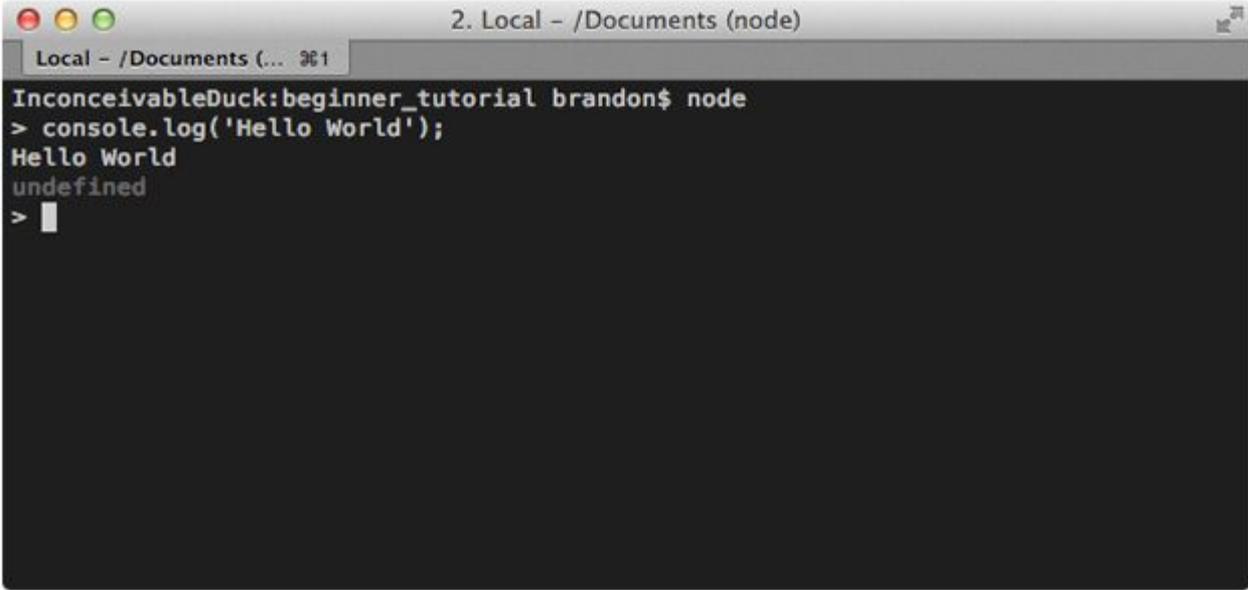
node processing model

- Официальный сайт находится по адресу <http://nodejs.org/>
- Hello world
- Сразу после установки вам становится доступна новая команда `node`. Её можно использовать двумя разными способами. Первый способ — без аргументов. Откроется интерактивная оболочка (REPL: `read-eval-print-loop`), где вы можете выполнять обычный JavaScript-код.



The screenshot shows the Node.js website homepage. At the top, there is a navigation bar with links for HOME, ABOUT, DOWNLOADS, DOCS, FOUNDATION, GET INVOLVED, SECURITY, and NEWS. Below the navigation bar, the Node.js logo is displayed. The main content area features a paragraph describing Node.js as a JavaScript runtime built on Chrome's V8 JavaScript engine, highlighting its event-driven, non-blocking I/O model and the npm package ecosystem. A green banner below the text reads "Important security releases, please update now!". Underneath, there are two buttons for downloading Node.js: "v4.4.0 LTS Recommended For Most Users" and "v5.8.0 Stable Latest Features". At the bottom, there are links for "Other Downloads | Changelog | API Docs" and a link to "Or have a look at the LTS schedule."

```
$ node
> console.log('Hello World');
Hello World
undefined
```

A screenshot of a terminal window on a Mac. The window title is "2. Local - /Documents (node)". The terminal shows the following text:

```
InconceivableDuck:beginner_tutorial brandon$ node
> console.log('Hello World');
Hello World
undefined
> █
```

# Куда попал node.js

ановка и запуск

C:\Program Files\nodejs

Name	Name
.	
node_modules	
node.exe	
node_etw_provider.man	
node_perfctr_provider.man	
nodevars.bat	
npm	
npm.cmd	

# Когда ставиться прописывается себя в переменную PATH

```
C:\Users\Anna>set PATH
Path=C:\Users\Anna\AppData\Roaming\npm;C:\Program Files\nodejs\;C:\ProgramData\Oracle\Java\javapath;C:\WINDOWS\system32;
C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\Users\Anna\.dnx\bin;C:\Program Files\
Microsoft DNX\Dnvm\;C:\Program Files\Microsoft SQL Server\130\DTS\Binn\;C:\Program Files\Microsoft SQL Server\Client SDK
\ODBC\110\Tools\Binn\;C:\Program Files (x86)\Microsoft SQL Server\130\Tools\Binn\;C:\Program Files\Microsoft SQL Server\
130\Tools\Binn\;C:\Program Files (x86)\Microsoft SQL Server\130\Tools\Binn\ManagementStudio\;C:\Program Files (x86)\Micr
osoft SQL Server\130\DTS\Binn\;C:\Program Files (x86)\Skype\Phone\;C:\Program Files (x86)\PICT\D:\Автоматизация тестиро
вания\apache-maven-3.3.9\bin;C:\Program Files\Java\jdk1.8.0_65\bin;C:\Program Files (x86)\Microsoft SQL Server\120\Tools
\Binn\;C:\Program Files\Microsoft SQL Server\120\Tools\Binn\;C:\Program Files\Microsoft SQL Server\120\DTS\Binn\;C:\Prog
ram Files (x86)\Microsoft SQL Server\120\Tools\Binn\ManagementStudio\;C:\Program Files (x86)\Microsoft SQL Server\120\DT
S\Binn\;C:\Program Files\nodejs\;C:\Users\Anna\AppData\Local\Programs\Python\Python35\Scripts\;C:\Users\Anna\AppData\Loc
al\Programs\Python\Python35\;C:\Users\Anna\AppData\Local\Programs\Python\Launcher\;C:\Users\Anna\AppData\Roaming\npm
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY
C:\Users\Anna>
```

Если команда node не выполняется то проверте переменную PAtH

# REPL – режим. Когда в командной строке выполняются операции.

```
C:\Users\Anna>node
> 1*7
7
> 5/8
0.625
> function f(a,b) {return a+b;}
undefined
> f(4,9)
13
>
```

Если нажать два раза ctrl+C мы выйдем из этого режима.

# Первая программа

1. На диске C создаем папку node в ней файл 1.js

```
console.log("Hello word")
```

В командной строке заходим в папку node

```
C:\node>node 1.js  
Hello word  
  
C:\node>
```

# Качаем дополнительно

Important **security releases**, please update now!

Download for Windows (x64)

**v4.4.0 LTS**

Recommended For Most Users

**v5.9.0 Stable**

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#) [Other Downloads](#) | [Changelog](#) | [API Docs](#)

Or have a look at the **LTS schedule**.

# Скачать Source Code

Распаковать в node js. Для просмотра и модернизации кода стандартных модулей.

Windows Installer (.msi)

Windows Binary (.exe)

Mac OS X Installer (.pkg)

Mac OS X Binaries (.tar.gz)

Linux Binaries (.tar.xz)

Source Code

32-bit	64-bit
32-bit	64-bit
64-bit	
64-bit	
32-bit	64-bit
<a href="#">node-v5.9.0.tar.gz</a>	

# Следующий шаг

1. Создаем папку `node` на диске `C`
2. Желательно создать проект в `RНР Shtorm` или `WebShtorm`
3. Настроить путем добавления плагина `nodejs` (`ctrl+alt+s`) установить плагин если его нет
4. В папке `node` создать папку примеров и в ней файл с расширением `js`

# Написать в нем код

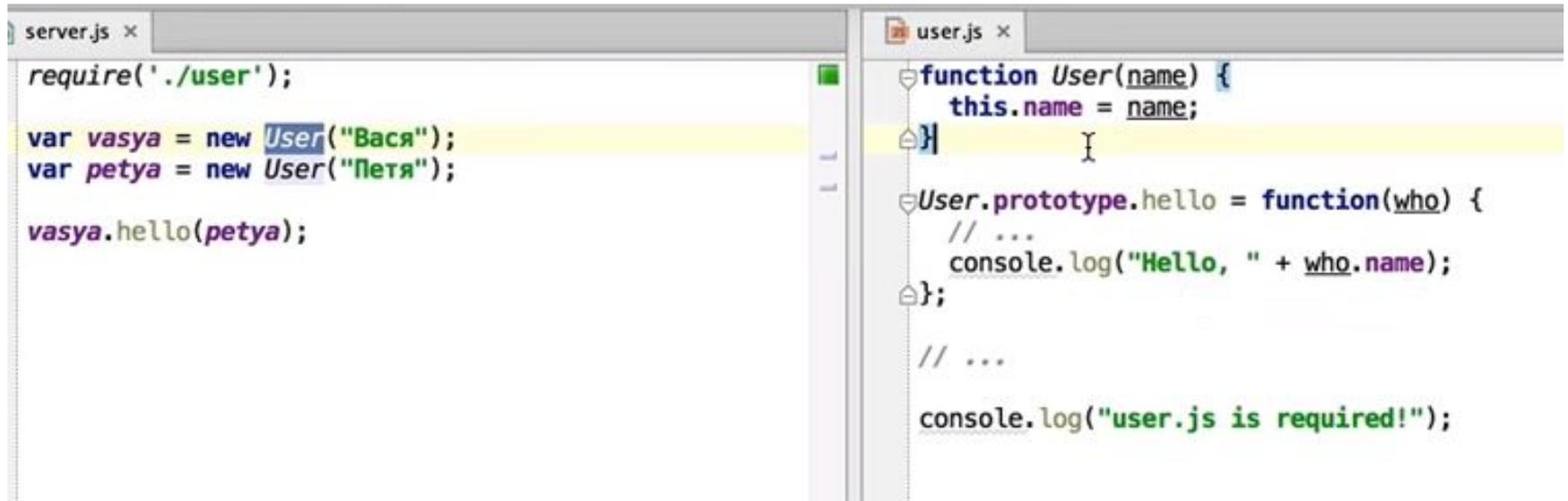
---

```
/** Created by Anna on 17.03.2016. ...*/  
function User(name) {  
    this.name=name;  
}  
User.prototype.goTohome = function(who) {  
    console.log("Go to home, "+who.name);  
};  
  
var anna = new User("Anna");  
var andrey = new User("andrey");  
  
anna.goTohome (andrey);
```

# Модули node.js

- Подключение модулей через команду `require (' name modul')`
- В `node.js` переменные одного модуля не видны в других модулях. Для видимости переменных используют переменную `exports`.

- Создать два файла



The image shows a code editor with two tabs: 'server.js' and 'user.js'. The 'server.js' tab is active and contains the following code:

```
require('./user');  
  
var vasya = new User("Вася");  
var petya = new User("Петя");  
  
vasya.hello(petya);
```

The 'user.js' tab is also visible and contains the following code:

```
function User(name) {  
  this.name = name;  
}  
  
User.prototype.hello = function(who) {  
  // ...  
  console.log("Hello, " + who.name);  
};  
  
// ...  
  
console.log("user.js is required!");
```

Запустить из консоли и получить ошибку что не видна переменная User. Используя exports починить ошибку.

```
server.js x user.js x
var user = require('./user');
var vasya = new user.User("Вася");
var petya = new user.User("Петя");
vasya.hello(petya);

// exports
function User(name) {
  this.name = name;
}
User.prototype.hello = function(who) {
  // ...
  console.log("Hello, " + who.name);
};
// ...
console.log("user.js is required!");
exports.User = User;
```

Global – глобальная переменная

```
server.js x user.js x
require('./user');
var vasya = new User("Вася");
var petya = new User("Петя");
vasya.hello(petya);

// exports
// global
function User(name) {
  this.name = name;
}
User.prototype.hello = function(who) {
  // ...
  console.log("Hello, " + who.name);
};
// ...
console.log("user.js is required!");
global.User = User;
```

# Пишем калькулятор

```
var ipAddress = "127.0.0.1";

// Defining the port on which we want to listen
var portNumber = "52000";

// Importing necessary library files
var httpModule = require("http");

// Creating our server's main method
httpModule.createServer(
function serviceRequest (request, response) {

    // Check what file the user has requested and take necessary action
    var queryString = new String(request.url);

    // We're expecting URLs of the following type:
    // action=add&number1=3&number2=6

    var keyValuePairs = queryString.split("&"); // Splitting the query string based on & delimiter

    // Now keyValuePairs[0] contains our action
    var action = keyValuePairs[0].replace("/", "").split("=")[1]; // extracting the action specified in the URL
    var firstNumber = new String(keyValuePairs[1].split("&").split("=")[1] || "0"); // extracting the first number
    var secondNumber = new String(keyValuePairs[2].split("&").split("=")[1] || "0"); // extracting the second number

    // calling the method to get the result
    var result = getResult(action.toLowerCase(), Number(firstNumber) , Number(secondNumber));

    // HTML which we will display to the user
    var htmlContent = "<html><b>" + action + "(" + firstNumber + "," + secondNumber + ") = <b>" + result + "</b></html>";

    // write the response
    response.end(htmlContent);
}
).listen(portNumber, ipAddress);
```

```
// Utility method to perform an operation on 2 numbers. Helps to modularize code
function getResult(operation, number1, number2)
{
    var result = 0;

    if(operation == "add")
    |   result = number1 + number2;

    else if(operation == "subtract")
    |   result = number1 - number2;

    else if(operation == "multiply")
    |   result = number1 * number2;

    else if(operation == "divide" && number2 != 0)
    |   result = number1 / number2;

    return result;
}
```

# После запуска файла из КОНСОЛИ

Запускаем браузер:

<http://127.0.0.1:52000/action=add&number1=18&number2=6>

[http://127.0.0.1:52000/action=divide&number1=18  
&number2=6](http://127.0.0.1:52000/action=divide&number1=18&number2=6)

<http://127.0.0.1:52000/action=multiply&number1=1034&number2=21>

<http://127.0.0.1:52000/action=subtract&number1=1034&number2=21>