How To Write

Node.js Module

vunb

Topics.

Node.js Modules

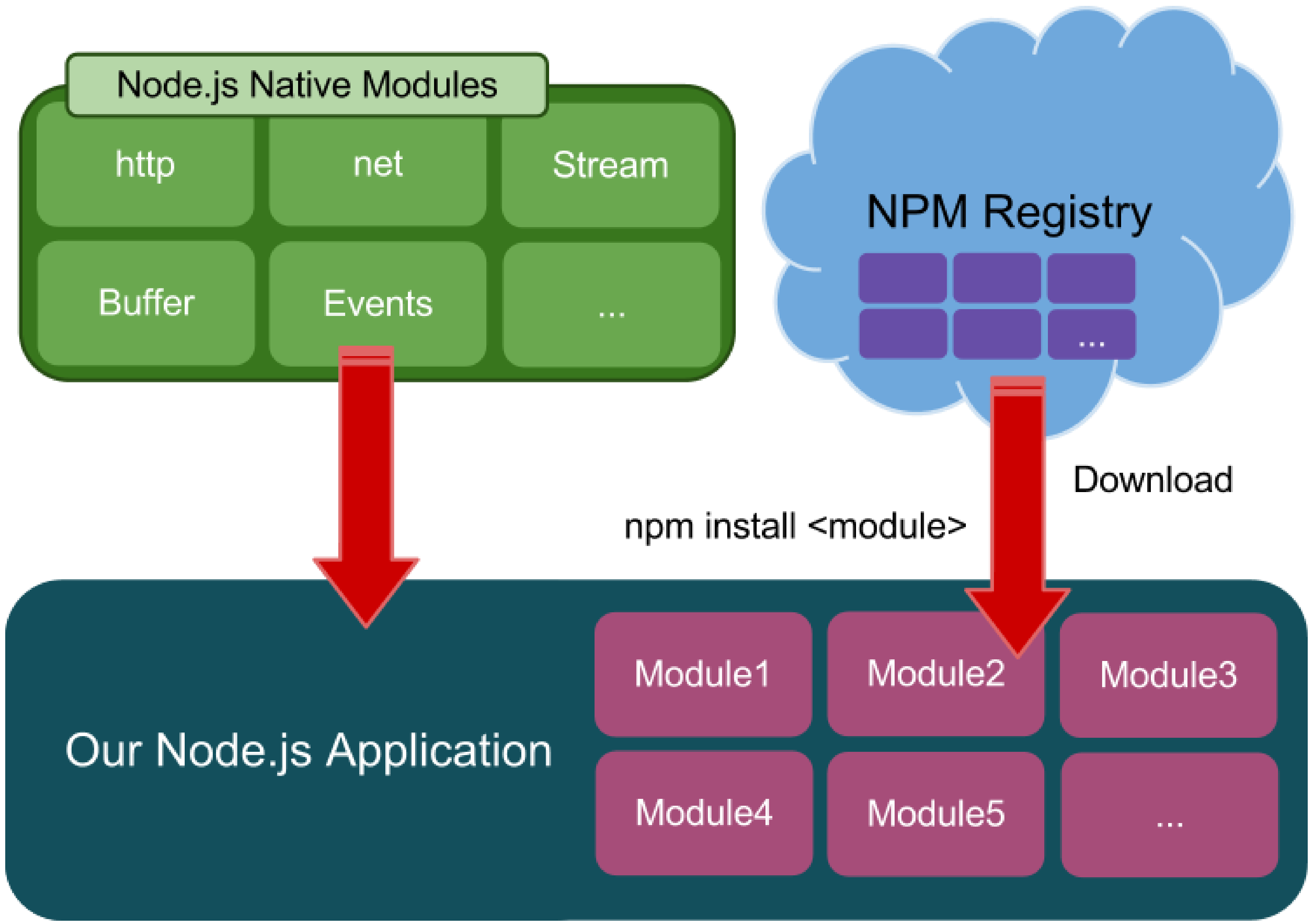
NPM Registry

C/C++ Addons

What is
Node.js Module ?

npm install *<something>*

You Must Be Familiar With


Node.js Native Modules
http
net
Stream
Buffer
Events
...
NPM Registry
...
Download
npm install <module>
Our Node.js Application
Module1
Module2
Module3
Module4
Module5
...

You Can Write Module In C/C++ & JavaScript

How to
Load Module

Load Global Module

Example:

var MyModule = require('mymodule');

Node.js will searching in the following location:

- *./node_modules*
- *../node_modules*
- *$HOME/.node_modules*
- *$HOME/.node_libraries*
- *$PREFIX/lib/node*

Load Local Module

Load the module in the same directory:

```
var MyModule = require('./mymodule');
```

Or

```
var MyModule = require('./mymodule.js');
```

Write The First
Node.js Module

The First Module Example

```
module.exports = function() {

    console.log('Hello World!');

};
```

require() ⟺ module.exports

Bridge between app and module

Implement a Class in Module

```
module.exports = function() {
    var self = this;
    this.counter = 0;

    this.pump = function() {
        self.counter++;
    };

};
```

More JavaScript Styles

```
var Pumper = module.exports = function() {
    this.counter = 0;
};

Pumper.prototype.pump = function() {
    Pumper.counter++;
};
```

Export Objects and Constants

```
var Pumper = module.exports.Pumper = function() {
    this.counter = 0;
};

Pumper.prototype.pump = function() {
    Pumper.counter++;
};

module.exports.Pumper1 = function() { ... };
module.exports.Pumper2 = function() { ... };
module.exports.Birthday = 714;
```

index.js
&
index.node

./example/index.js

```
var ex = require('./example');
```

Let's
Publish Your Module

NPM Registry

NPM = Node Package Manager

NPM Registry

npmjs.org

Steps to Publish Package

1. Get **NPM Account**
2. Generate **package.json**
3. To **Upload** Package

Get NPM Account

npm adduser

Initialize Package

npm init

Output: package.json

Run "npm init"

```
$ npm init
Package name: (demo)
Description: Hello
Package version: (0.0.0)
Project homepage: (none)
Project git repository: (none)
Author name: vunb
Author email: (none) vunb@nodejs.vn
Author url: (none)
Main module/entry point: (none)
Test command: (none)
```

We got package.json

```
{
  "author": "vunb <vunb@nodejs.vn>",
  "name": "demo",
  "description": "Hello",
  "version": "0.0.0",
  "repository": {
    "url": ""
  },
  "dependencies": {},
  "devDependencies": {},
  "optionalDependencies": {},
  "engines": {
    "node": "*"
  }
}
```

Normal Structure of Package

- index.js
- package.json
- README (README.md)
- LICENSE
- lib/hello1.js
- lib/hello2.js
- test/test1.js
- test/test2.js

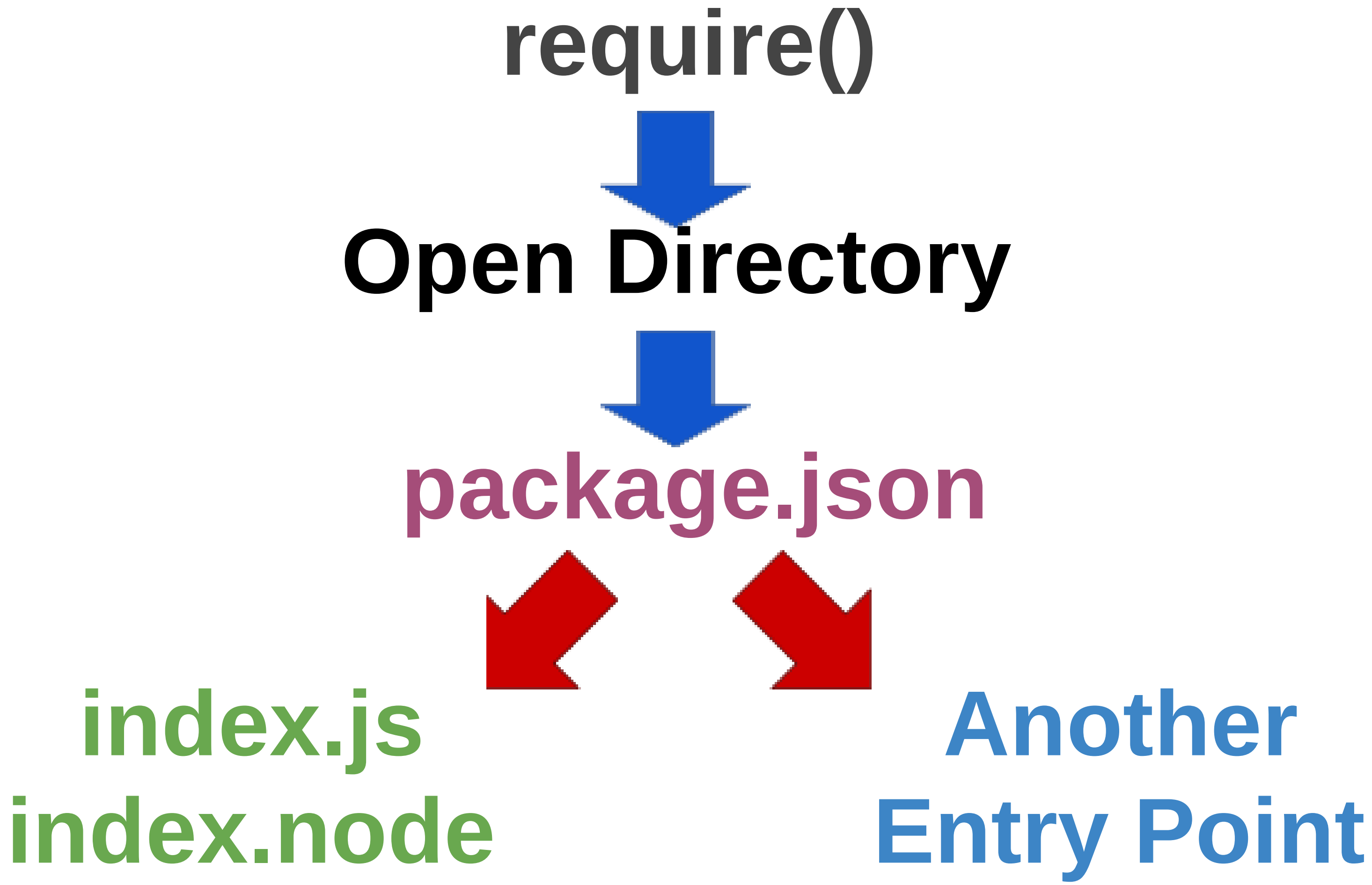
I don't want to use index.js !

Change Entry Point

Add "**main**" Property To **package.json**

After Change Entry Point

- demo.js
- package.json
- README (README.md)
- LICENSE
- lib/hello1.js
- lib/hello2.js
- tests/test1.js
- tests/test2.js


require()
Open Directory
package.json
index.js
index.node
Another
Entry Point

Upload Package

npm publish .

C++

Advanced Topic

How to
Write C/C++ Addons

Development Environment

1. **GCC** (used to compile)
2. **Python** (For build script)

Write The First
C/C++ Addon

C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
    HandleScope scope;
    return scope.Close(String::New("world"));
}

void init(Handle<Object> target) {
    target->Set(String::NewSymbol("hello"),
        FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
    HandleScope scope;
    return scope.Close(String::New("world"));
}

void init(Handle<Object> target) {
   target->Set(String::NewSymbol("hello"),
       FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

module.exports

C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
    HandleScope scope;
    return scope.Close(String::New("world"));
}

void init(Handle<Object> target) {
    target->Set(String::NewSymbol("hello"),
        FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

```
function() {
    return 'world';
}
```

Compare with JavaScript Version

```
var target = module.exports;

target['hello'] = function() {
    return 'world!';
};
```

Or

```
module.exports.hello = function() {
    return 'world!';
};
```

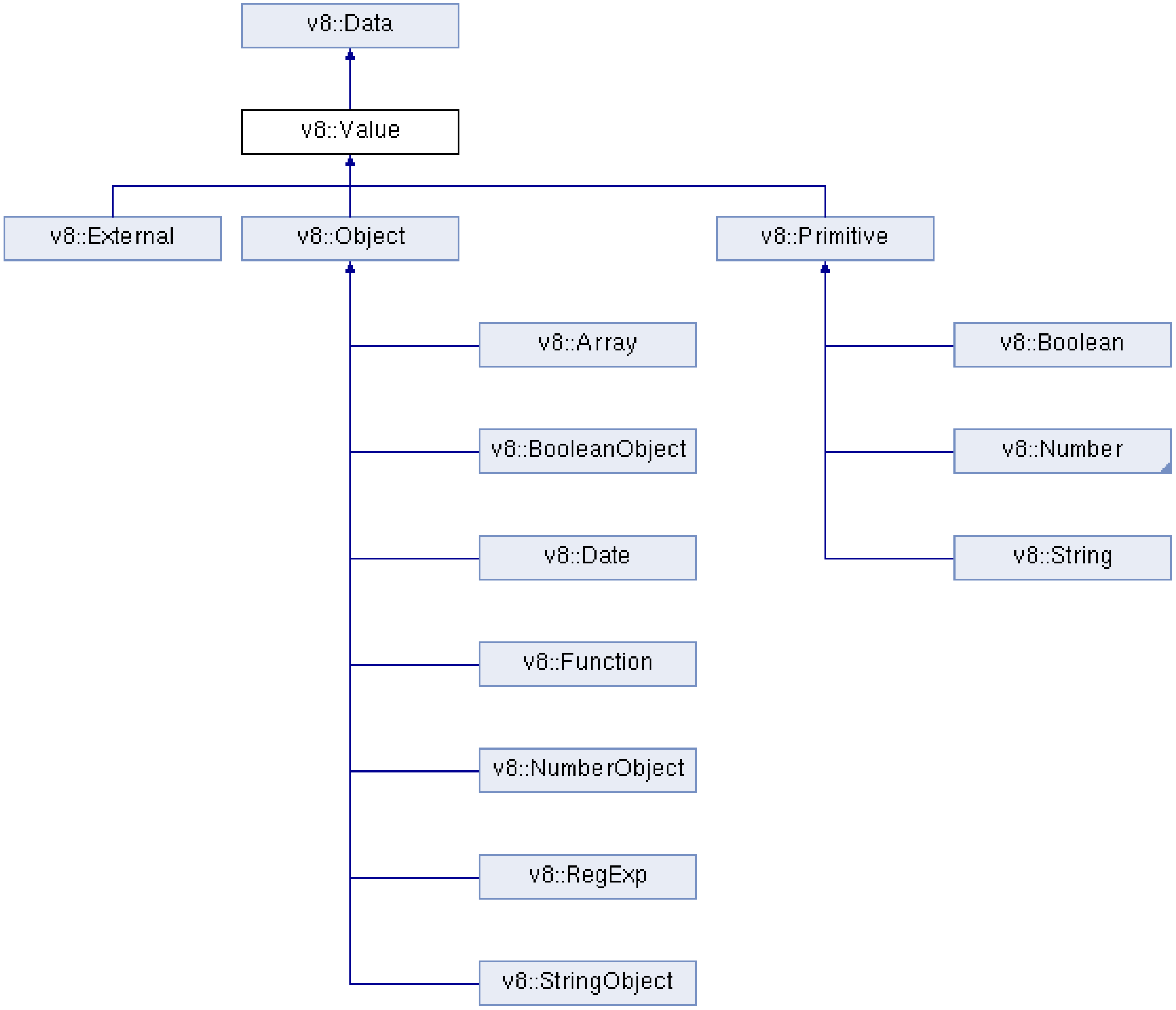
C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
    HandleScope scope;
    return scope.Close(String::New("world"))
}

void init(Handle<Object> target) {
    target->Set(String::NewSymbol("hello"),
        FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
   HandleScope scope;
   return scope.Close(String::New("world"));
}

void init(Handle<Object> target) {
   target->Set(String::NewSymbol("hello"),
      FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

v8::String Class


v8::Data
v8::Value
v8::External
v8::Object
v8::Primitive
v8::Array
v8::BooleanObject
v8::Date
v8::Function
v8::NumberObject
v8::RegExp
v8::StringObject
v8::Boolean
v8::Number
v8::String

C/C++ Addon Example

```
#include <node.h>
#include <v8.h>

using namespace v8;

Handle<Value> Method(const Arguments& args) {
    HandleScope scope;
    return scope.Close(String::New("world"));
}

void init(Handle<Object> target) {
    target->Set(String::NewSymbol("hello"),
        FunctionTemplate::New(Method)->GetFunction());
}

NODE_MODULE(hello, init);
```

HandleScope

- Determine Lifetime of handles
- Often created at the beginning of a function call
- Deleted after function return
    - **scope.Close(*<Handle>*)** to avoid handle being deleted by **Garbage Collection**

Compile The First
C/C++ Addon