등업소스 보관함 >

남들이 안갖고 저만의 require 소스.[퍼블릭 도메인 공유]

jomin398 챗봇 마스터 1:1 채팅
2020.04.01. 00:59 조회 106

본소스는 public Domain 소스입니다.
Michael Mathews. 라는 깃헙 유저가 작성하셨고
전 이분 소스 빌려왔었습니다.

require 의 사용방법이면
node.js 랑 같은 방식이지만.
그냥 파일 주소를 적으면 되요.
sdcard 내부 아무대나...
index.js 를 가리키면되요..

그런 초고급 소스를 공유합니다!

```
/*
Rhino-Require is Public Domain
Written by Michael Mathews. Licensed as public domain.
<http://en.wikipedia.org/wiki/Public_Domain>

The author or authors of this code dedicate any and all copyright interest
in this code to the public domain. We make this dedication for the benefit
of the public at large and to the detriment of our heirs and successors. We
intend this dedication to be an overt act of relinquishment in perpetuity of
all present and future rights to this code under copyright law.
*/

(function(global) {

var require = global.require = function(id) { /*debug*///console.log('require('+id+')');
if (typeof arguments[0] !== 'string') throw 'USAGE: require(moduleId)';

var moduleContent = '',
moduleUri;

moduleUri = require.resolve(id);
moduleContent = '';

if (require.cache[moduleUri]) {
return require.cache[moduleUri];
}

var file = new java.io.File(moduleUri);
try {
var scanner = new java.util.Scanner(file).useDelimiter("₩₩Z");
moduleContent = String( scanner.next() );
}
catch(e) {
throw 'Unable to read file at: '+moduleUri+', '+e;
```

```
}

if (moduleContent) {
try {
var f = new Function('require', 'exports', 'module', moduleContent),
exports = require.cache[moduleUri] || {},
module = { id: id, uri: moduleUri, exports: exports };


require._root.unshift(moduleUri);
f.call({}, require, exports, module);
require._root.shift();
}
catch(e) {
throw 'Unable to require source code from "' + moduleUri + '": ' + e.toSource();
}

exports = module.exports || exports;
require.cache[moduleUri] = exports;
}
else {
throw 'The requested module cannot be returned: no content for id: "' + id + '" in paths: ' + require.paths.join(', ');
}

return exports;
}
require._root = [''];
require.paths = [];
require.cache = {}; // cache module exports. Like: {id: exported}

var SLASH = Packages.java.io.File.separator;

/** Given a module id, try to find the path to the associated module.
*/
require.resolve = function(id) {
// TODO: 1. load node core modules

// 2. dot-relative module id, like './foo/bar'
var parts = id.match(/^(₩.?₩.(?:₩₩|₩/)|(?:₩₩|₩/))(.+)$/),
isRelative = false,
isAbsolute = false,
basename = id;

if (parts) {
isRelative = parts[1] === './' || parts[1] === '.₩₩' || parts[1] === '../' || parts[1] === '..₩₩';
isAbsolute = parts[1] === '/' || parts[1] === '₩₩';
basename = parts[2];
}

if (typeof basename !== 'undefined') {

if (isAbsolute) {
rootedId = id;
}
else {
```

```
else {
var root = (isRelative? toDir(require._root[0] || '.') : '.'),
rootedId = deDotPath(root + SLASH + id),
uri = '';
}

if ( uri = loadAsFile(rootedId) ) { }
else if ( uri = loadAsDir(rootedId) ) { }
else if ( uri = loadNodeModules(rootedId) ) { }
else if ( uri = nodeModulesPaths(rootedId, 'rhino_modules') ) { }
else if ( uri = nodeModulesPaths(rootedId, 'node_modules') ) { }

if (uri !== '') return toAbsolute(uri);

throw 'Require Error: Not found.';
}
}

/** Given a path, return the base directory of that path.
@example toDir('/foo/bar/somefile.js'); => '/foo/bar'
*/
function toDir(path) {
var file = new java.io.File(path);

if (file.isDirectory()) {
return path;
}

var parts = path.split(/[\\/]/);
parts.pop();
return parts.join(SLASH);
}

/** Returns true if the given path exists and is a file.
*/
function isFile(path) {
var file = new java.io.File(path);

if (file.isFile()) {
return true;
}

return false;
}

/** Returns true if the given path exists and is a directory.
*/
function isDir(path) {
var file = new java.io.File(path);

if (file.isDirectory()) {
return true;
}

return false;
```

```
}

/** Get the path of the current working directory
*/
function getCwd() {
return deDotPath( toDir( ''+new java.io.File('.').getAbsolutePath() ) );
}

function toAbsolute(relPath) {
absPath = ''+new java.io.File(relPath).getAbsolutePath();
absPath = deDotPath(absPath);
return absPath;
}

function deDotPath(path) {
return String(path)
.replace(/(₩/|₩₩)[^₩/₩₩]+₩/₩.₩.(₩/|₩₩)/g, SLASH)
.replace(/(₩/|₩₩)₩.(₩/|₩₩|$)/g, SLASH);
}

/** Assume the id is a file, try to find it.
*/
function loadAsFile(id) {
if ( isFile(id) ) { return id; }

if ( isFile(id+'.js') ) { return id+'.js'; }

if ( isFile(id+'.node') ) { throw 'Require Error: .node files not supported'; }
}

/** Assume the id is a directory, try to find a module file within it.
*/
function loadAsDir(id) {
if (!isDir(id)) {
return;
}
// look for the "main" property of the package.json file
if ( isFile(id+SLASH+'package.json') ) {
var packageJson = readFileSync(id+SLASH+'package.json', 'utf-8');
eval( 'packageJson = '+ packageJson);
if (packageJson.hasOwnProperty('main')) {
var main = deDotPath(id + SLASH + packageJson.main);
return require.resolve(main);
}
}

if ( isFile(id+SLASH+'index.js') ) {
return id+SLASH+'index.js';
}
}

function loadNodeModules(id) {
var path,
uri,
cwd = getCwd()
```

```
cwd = getCwd(),
dirs = cwd.split(SLASH),
dir = dirs.join(SLASH);
for (var i = 0, len = require.paths.length; i < len; i++) {
path = require.paths[i];
path = deDotPath(dir + SLASH + path);
if (isDir(path)) {
path = deDotPath(path + SLASH + id);

uri = loadAsFile(path);
if (typeof uri !== 'undefined') {
return uri;
}

uri = loadAsDir(path);
if (typeof uri !== 'undefined') {
return uri;
}
}
}
}

function nodeModulesPaths(id, moduleFolder) {
var cwd = getCwd(),
dirs = cwd.split(SLASH),
dir,
path,
filename,
uri;

while (dirs.length) {
dir = dirs.join(SLASH);
path = dir+SLASH+moduleFolder;

if ( isDir(path) ) {
filename = deDotPath(path+SLASH+id);

if ( uri = loadAsFile(filename) ) {
uri = uri.replace(cwd, '.');
return uri;
}

if ( uri = loadAsDir(filename) ) {
uri = uri.replace(cwd, '.');
return uri;
}
}

dirs.pop();
}
}

function readFileSync(filename, encoding, callback) {
if (typeof arguments[1] === 'function') {
encoding = null;
```

```
callback = arguments[1];
}

encoding = encoding || java.lang.System.getProperty('file.encoding');

try {
var content = new java.util.Scanner(
new java.io.File(filename),
encoding
).useDelimiter("₩₩Z");

return String( content.next() );
}
catch (e) {
return '';
}
}

})(this);
```


jomin398님의 게시글 더보기 >

좋아요 **1** 댓글 **2** 공유 | 신고

댓글 등록순 최신순

성빈
이게머시여
2020.04.01. 01:06

jomin398 작성자
ㅋㅋ require 인거에요
그것도 파일 경로로 가능해요
2020.04.01. 01:23

글쓰기 답글 목록 ▲ TOP

'등업소스 보관함' 게시판 글

이 게시판 새글 구독하기

엄 이렇게 쓰는 게 맞나요 [6]	윤서는뽀뽀충	2020.04.01.
배터리 온도 알려주는 아이 [2]	루카	2020.04.01.
남들이 안갖고 저만의 require 소스.[퍼블릭 도메인 공유] [2]	jomin398	2020.04.01.
[소스] 엄청 짧은 인사! [2]	AlphaDo	2020.04.01.
[소스] 엔터 -> ₩n 로 바꿔주는 소스 [2]	Alpha	2020.04.01.

1 2 3 전체보기

이 카페 인기글

[치***, 한**] 님 강제 탈퇴

AlphaDo
♡1 ⋯2

공식 카카오톡 옾챗

Lunar
♡0 ⋯9

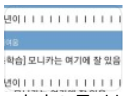
혹시 여기서 변수를 저장하는 부분이 어딘지 아시는분 계시나요..??
마녀사냥 금지

Milk2
AlphaDo
♡3 ⋯8

예전에 어떤분이 카톡봇을 이용하여 카페에 게시글을 올렸었는데

리페
♡0 ⋯1

"시초가 중국발" 발언 관련 해명글 및 부탁의 글



[카카오톡 봇] 양산형 자동학습 봇 소스 공유

함수를 입력받아 그래프 그리기

트위치봇
Steve28
♡2 ⋯4

네이버에서 지역별 날씨 정보 크롤링하기 /

1 2 3 4 5