Rapid Language

Guide（教程）

Index 索引：

Introduction 介绍

Rapid is a programming language based on Node.js.
Rapid 是一个基于 Node.js 的编程语言。

It's a superset of Node.js, very fast, and the syntax is very simple.
它是一个 Node.js 的超集，速度非常快，语法非常简单。

Any Rapid code can be compiled into JavaScript with good performance.
任何 Rapid 代码都可以被编译成为 JavaScript，同时有着良好的性能。

Before learning Rapid, you need to understand some basic JavaScript and Node.js syntax. If you don't already know, please visit:
https://developer.mozilla.org/docs/Web/JavaScript
And:
https://nodejs.org/en/docs/

学习 Rapid 之前需要你了解一些基础的 JavaScript、Node.js 语法。如果你还没有了解，请访问：
https://developer.mozilla.org/docs/Web/JavaScript
和：
https://nodejs.org/en/docs/

Rapid Official website: https://18510047382.github.io/rapid-website/ **(URL will be changed in the future)**
Rapid 网址：https://18510047382.github.io/rapid-website/ **（网址在将来将会更改）**

Command 命令

Rapid command currently has the following parameters:
Rapid 命令目前具有以下几个参数：

- -v | -version 输出 Rapid 版本信息 Output Rapid version information
- -e | -execute [filepath] 执行 Rapid 代码 Execute Rapid code
- -c | -compile [filepath] 编译 Rapid 代码，并存储到同名 js 文件里
  Compile the Rapid code and store it in the same name js file

直接使用 rapid 命令不加任何参数可以在控制台中打开实时 Rapid 代码调试台。
You can open the real-time Rapid code debug console in the console without using any parameters directly using the rapid command.

例如 Example：

$ rapid -v

Path Problem 路径问题

与 Node.js 不同的是，你没有办法使用 ./ 来确定项目位置，你只能使用 module.__dirname
Unlike Node.js, you have no way to use ./ to determine the project location. You can only use module.__dirname

例如 Example

module.__dirname + filepath

```
module.__dirname + myFilePath
```

但是当你使用 module.require 和 #import 语法时，你可以使用 ./ 来表示相对路径，因为 Rapid 会自动转换它为绝对路径。除此之外，你**不**应该使用 ./ 来表示相对路径。

But when you use the module.require and #import syntax, you can use ./ to represent a relative path, because Rapid will automatically convert it to an absolute path. Other than that, you should **not** use ./ to indicate relative paths.

Console Output 控制台输出

在 Node.js 中，你可以使用 console.log 、 console.error 和 console.warn 在控制台中输出内容。
当时在 Rapid 中，你可以使用关键字 print、error 和 warn 来输出内容。

In Node.js, you can use console.log , console.error , and console.warn to output content in the console.
At Rapid, you could use the keywords print, error, and warn to output content.

例如 Example

print ‘hello world’
error ‘hello world’
warn ‘hello world’

```
print 'Hello World';
warn 'Oh warning!';
error 'Got Some Error!';
```

Declare variable 变量声明

你可以使用 **let** 和 **const** 来声明变量和常量，但是不能使用 var

You can use **let** and **const** to declare variables and constants, but you can't use var

```
let msg = 'Hello';
const VERSION = 1.0;
```

（在当前的 Rapid 版本中，你还是可以使用 var 声明变量，但是你会收到来自控制台的警告 !）

(In the current Rapid release, you can still declare variables using var , but you will receive a warning from the console!)

```
Rapid: WARN! You are using a keyword that does not receive support: var
Rapid: WARN! ------------------------------------------------------------
Rapid: WARN! The var keyword will be completely unsupported in later versions.
```

Declare function 函数声明

声明函数的方式略有不同，你可以使用关键字 func 来声明一个函数。
The way you declare a function is slightly different. You can use the keyword func to declare a function.

使用关键字 afunc 声明一个异步执行函数，相当于 async function
Declare an asynchronous execution function with the keyword afunc, which is equivalent to async function

```
func init() {
    print 'Welcome use Rapid!';
}
afunc init() {
    print 'Welcome use Rapid!';
}
```

用 function 声明函数的方式也是可行的，但是你依旧会收到来自控制台的警告！
The way you declare a function with function is also possible, but you will still receive a warning from the console!

Module 模块化

Rapid 内置了 module 对象，此对象内置字段如下。
Rapid has a built-in module object, the built-in fields of this object are as follows.

- __dirname 【项目路径 Project path】
- __filename【项目文件路径 Project file path】
- require(filepath)【导入一个文件，支持 node.js 内置模块 Import a file, support the built-in module of node.js】【需要使用 import 关键字来导入】
- export【导出部分内容 Export some thing】

使用方式 Example：

```
//Export some variable
module.export = {
    test: 'Hello'
}

//Import fs module
import fs = module.require('fs');

//Import Other module
import myModule = module.require('./myModule.rapid');
```

Rapid 不只提供了 module.require 函数导入模块，Rapid 还支持 #import 关键字来导入。
Rapid does not only provide the module.require function import module, Rapid also supports the #import keyword to import.

```
#import ./myModule.rapid
```

#import 导入的模块引用一次加载一次，require 导入的模块第一次引用会先将结果加载到内存中，之后的引用不再加载文件。
The #import imported module reference is loaded once, and the first reference of

the module imported by require will first load the result into memory, and the subsequent reference will no longer load the file.

同时，当你使用 -c 指令来编译一个 rapid 文件时，使用#import 导入的内容会直接被包含在 js 文件里，而 require 就不会这样。
Also, when you compile a rapid file with the -c command, the contents imported using #import will be directly included in the js file, and require will not.

Gui application 视图应用

你可以使用-c 指令来将 rapid 文件编译成 js 文件。同时使用 electron 加载这个 js 文件。

You can use the -c command to compile the rapid file into a js file. Also use electron to load this js file.

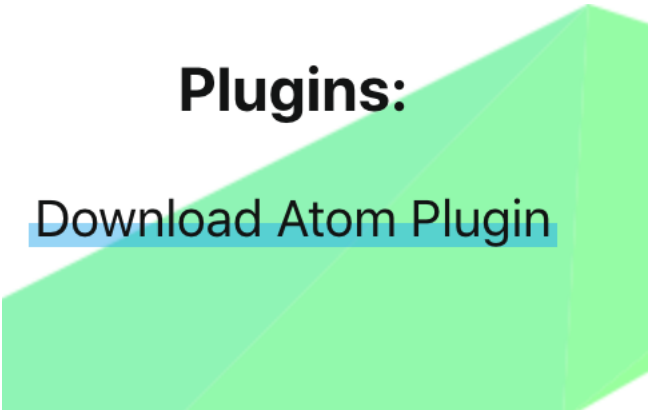
具体如何操作这里不提供说明，详情请参阅 electron 官网。

How to do this is not provided here. For details, please refer to the official website of electron.

Use Rapid in Atom 在 Atom 中使用 Rapid

想要在 atom 中获得语法高亮提示，你需要安装 atom-rapid-language 插件。
To get syntax highlighting in the atom, you need to install the atom-rapid-language plugin.

你可以在官网的下载界面上找到它。
You can find it on the official website's download interface.



下载完成之后解压它，并将它放到系统根目录下的 .atom 文件夹里的 package 文件夹里。
After the download is complete, unzip it and place it in the package folder in the .atom folder under the system root directory.