eggjs

新建项目

npm init egg –type=simple

npm i

# 启动项目

npm run dev

目录结构

egg-project

|── package.json

|── app.js (可选)

|── agent.js (可选)

|── app

| |── router.js(路由配置)

| |── controller(controller层，app接口)

| | |——home.js

| |── service (service层，供controller层调用，处理数据，可选)

| | |——user.js

| |── middleware (可选)

| | |——response\_time.js

| |── schedule (可选)

| | |——my\_task.js

| |── public (可选，静态资源)

| | |——reset.css

| |── view (可选)

| | |——home.tpl

| |——extend (可选)

| |── helper.js (可选)

| |── request.js (可选)

| |── response.js (可选)

| |── context.js (可选)

| |── application.js (可选)

| |——agent.js (可选)

|── config

| |── plugin.js

| |── config.default.js

| |── config.prod.js

| |── config.test.js (可选)

| |── config.local.js (可选)

| |—— config.unittest.js (可选)

|———— test

|── middleware

| |—— response\_time.test.js

|———controller

|—— home.test.js

http访问->router->controller->service

router路由

'use strict';

module.exports = app => {

const { router, controller } = app;

router.get('/', controller.home.index);

router.get('/hello',controller.hello.index);

};

router支持get，post，put，delete,

controller接口层

const Controller = require('egg').Controller;

class Hello extends Controller {

async index(){

let {service} = this.ctx;

…

this.ctx.body = …;

}

}

service服务层

const Service = require('egg').Service

class helloService extends Service {

async hello(){

return `hello service`;

}

}

module.exports = helloService;