Серверный Javascript

# Javascript на сервере

## Основные сведения

Для выполнения кода **javascript** на сервере используется библиотека **JInt** от Себастьяна Роса (<https://github.com/sebastienros/jint>). В ней реализована практически полная поддержка **ES6** за исключением некоторых специфических особенностей.

## Вызов javascript

Серверный javascript определяется типом команды "type":"javascript" в разделе commands файла model.json. В команде задается имя файла и произвольные аргументы, которые будут переданы в функцию **javascript**.

"commands": {  
 "testjs": {  
 "type": "javascript",  
 "file": "server.module", /\*имя модуля в текущем каталоге \*/  
 "args": {}, /\*произвольный объект с аргументами для скрипта \*/  
 }  
}

## Формат модуля

Модуль должен экспортировать единственную функцию, которая и будет вызвана.

/\*server module function\*/

module.exports = function (prms, args) {

//this: environment  
 //prms: parameters from client (object)

//args: arguments from model.json (object)

return {};

}

Значение (объект) которое вернет эта функция будет просто просто возвращено на клиент.

## Объект **environment**

Окружение (environment) позволяет обращаться к конфигурационной информации (из файла web.config), а также содержит функции для выполнения различных операций, таких как обращение к внешним источникам по протоколу http (fetch) и работу с моделями данных (loadModel, saveModel, executeSql).

При выполнении серверного кода окружение передается через контекст выполнения функции (this).

## Описание **environment**

export interface IConfig {

appSettings(key: string): object;

}

declare type execSqlParams = {

procedure: string,

source?: string,

parameters?: object

}

declare type saveModelParams = {

procedure: string,

data: object,

source?: string,

parameters?: object

}

declare type fetchParams = {

method: string;

body: object | string;

headers: object;

}

declare type fetchResponse = {

ok: boolean;

status: number;

statusText: string;

contentType: string;

body: string;

headers: object;

isJson(): boolean;

text(): string;

json(): string;

}

export interface IEnvironment {

config: IConfig;

loadModel(prms: execSqlParams): object;

saveModel(prms: saveModelParams): object;

executeSql(prms: execSqlParams): object;

fetch(url: string, prms: fetchParams): fetchResponse;

}

## Объект **config**

Объект config имеет единственный метод appSettings, который возвращает объект конфигурации из раздела <appSettings> файла web.config.

Например, если в web.config есть такой параметр:

<appSettings>  
 <add key="myApi" value="{param1:'value1', param2: 222, param3:true}" />  
</appSettings>

то вызов this.config.appSettings('myApi') вернет соответствующий объект со свойствами