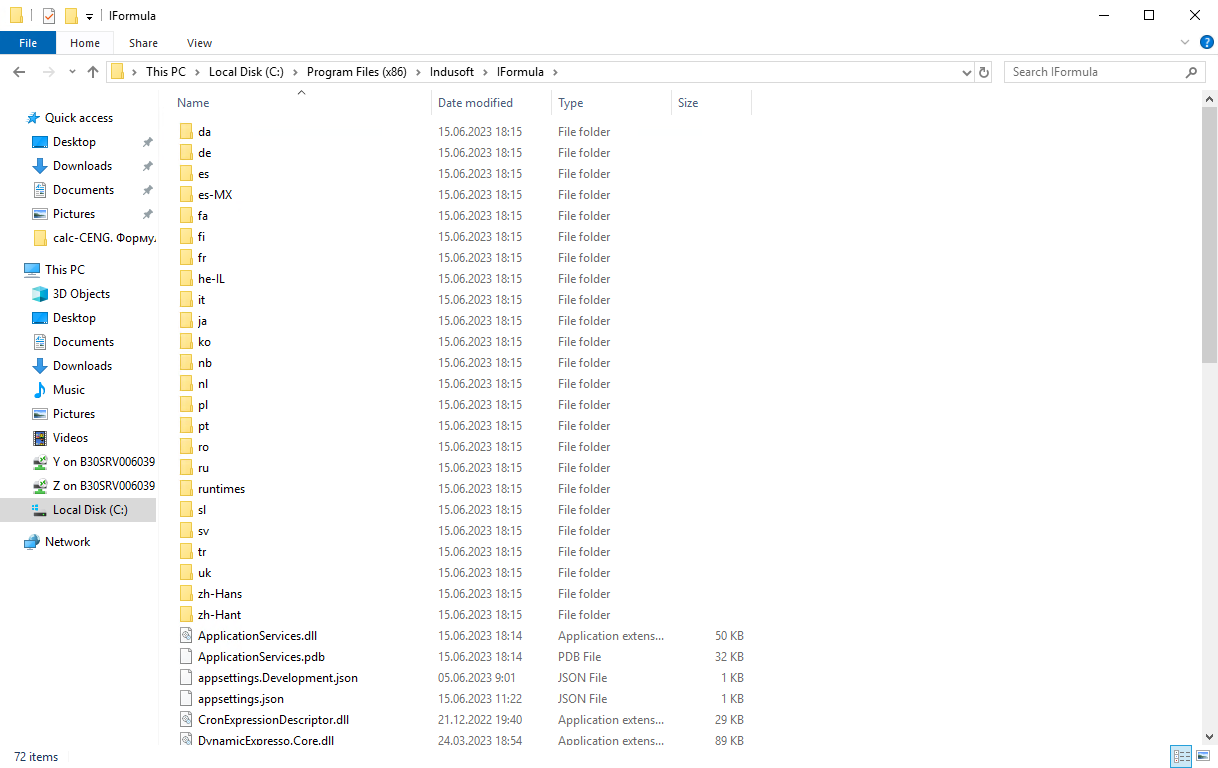
# Запуск приложения на сервере

Архив с приложением распаковываем на сервере:



## Настройка файлов конфигурации

### appsettings.json

DefaultConnection- строка подключения к БД I-DS\P,

TSDBAddress- адрес Tsdb WebApi,

TSDBLogin- логин Tsdb WebApi,

TSDBPassword- пароль Tsdb WebApi.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

### NLog.config

В секции rules можем выбрать/ добавить правила записи определённых логов конкретного уровня:

ApplicationServices.Calculator- логи расчётов,

ApplicationServices.Services и IFormula- логи приложения,

ApplicationServices.Scheduller – логи планировщика (по расписанию и триггеру).

В секции targets указаны места, куда записываются логи из соответствующих rules, а так же правила архивирования:

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

### Nodes.json

Данным конфигурационным файлом настраивается перечень узлов расчётов (можно добавлять необходимое количество через запятую):

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

SearchAttribute – атрибут, по которому в узел добавляется элемент, в который вложены расчётные атрибуты и значение которого true:

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание,

SearchModel – модель, в которой ищутся элементы ( \* - все модели),

SearchTemplate –шаблон элементов, которому должны соответствовать элементы ( \* - все шаблоны),

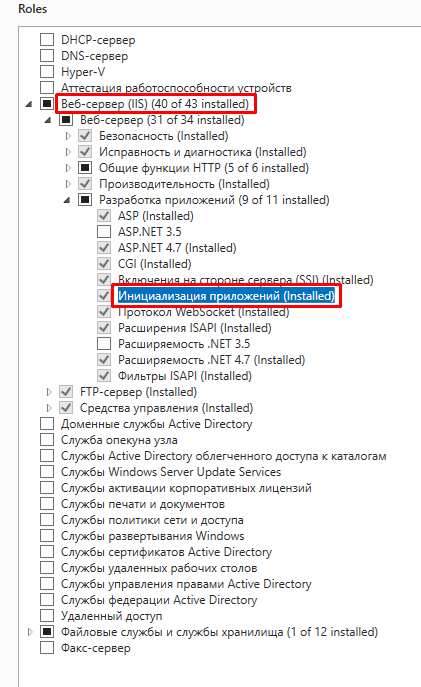
cronExpression – выражение планировщика по расписанию:  
<minute> <hour> <day-of-month> <month> <day-of-week> <command>

Более подробная информация о cron по ссылке:  
<https://losst.pro/nastrojka-cron>

https://www.baeldung.com/cron-expressions  
  
Пример: "0 0/1 \* ? \* \*" означает каждая минута в 0 секунд.

## 1.2 Настройка окружения

На сервере должна быть добавлена роль Веб-сервер IIS, а так же отдельно Инициализация приложений:



Должен быть установлен компонент .NET Core dotnet-hosting версии 6.0.15.

В IIS Manager во вкладке Application Pools необходимо добавить новый пул с соответствующими настройками:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Далее нажимаем Advanced Settings пула:

StartMode должен быть Always Running, Idle Time-out= 0 и Identity должна быть УЗ с правами к БД I-DS\P:

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Во вкладке Sites, выбрать Add Website, ввести название приложения, выбрать пул, указать каталог распакованного приложения и порт (8084 по умолчанию):

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

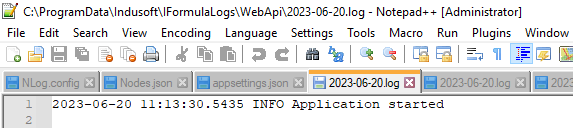
Автоматически созданное описание

В дополнительных настройках сайтах выставить значение True у параметра preloadEnabled:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

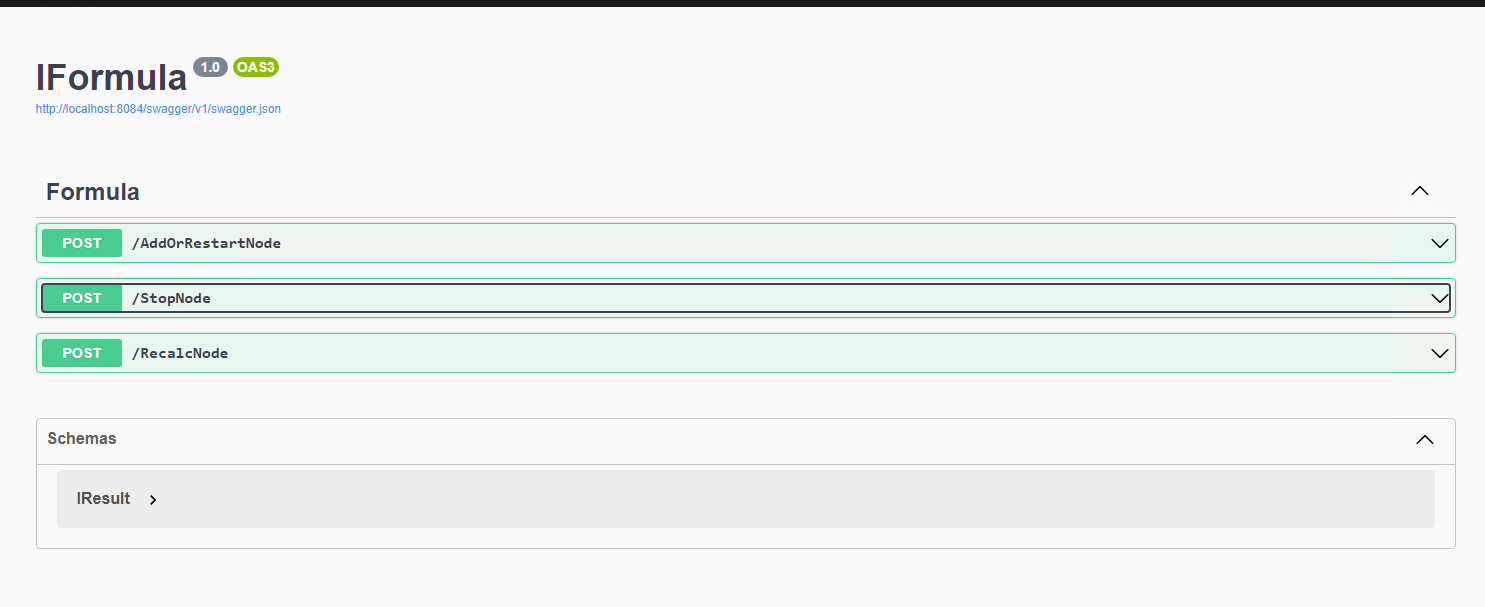
Автоматически созданное описание

Запустить пул приложения. При успешном запуске должна отобразиться следующая запись в логе WebApi:



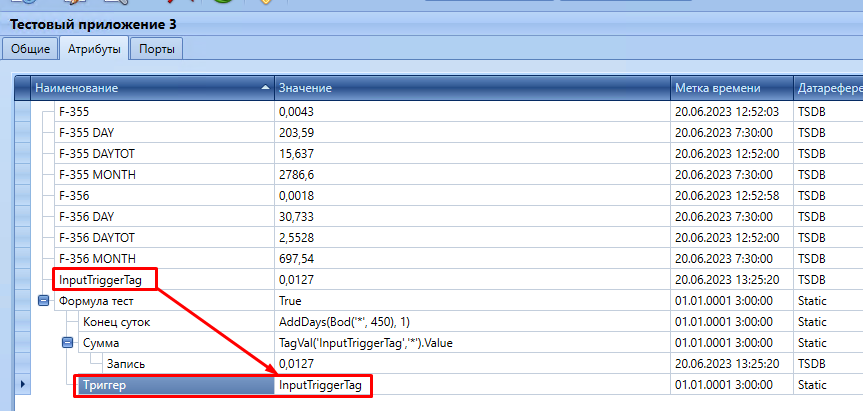
# Управление

Управлять можно из браузера через интерфейс Swagger’а, а так же любым Http клиентом:  
Например локально перейдя по ссылке <http://localhost:8084/swagger/index.html>

В н.в. имеются 2 контроллера: AddOrRestart, StopNode и RecalcNode:  


## 2.1 Запуск по триггеру

Для запуска по триггеру в элементе необходимо добавить вложенный static атрибут “Триггер” типа текст (внутри искомого атрибута SearchAttribute из Nodes.json), в значение через запятую необходимо прописать имя внешнего атрибута с тегом TSDB, по приходу значения которого будет запускаться расчёт элемента с соответствующей меткой времени:



При запуске узла расчётов с таким элементом будет указано количество элементов, запускаемых по триггеру и на соответствующие теги будет оформлена подписка на обновление значений в TSDB WebApi:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

При приходе значения в логах планировщика в каталоге TimersAndTriggers\Trigger будет запись нового значения триггерного тега, на метку которого запускается расчёт соответствующего элемента:

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, программное обеспечение, веб-страница, Веб-сайт

Автоматически созданное описание

## 2.2 Перезапуск узла расчётов (добавление или перезапуск)

Выбрать контроллер AddOrRestart, нажать Try it out, в поле Name необходимо записать имя атрибута узла расчётов из файла конфигурации Nodes.json приложения:

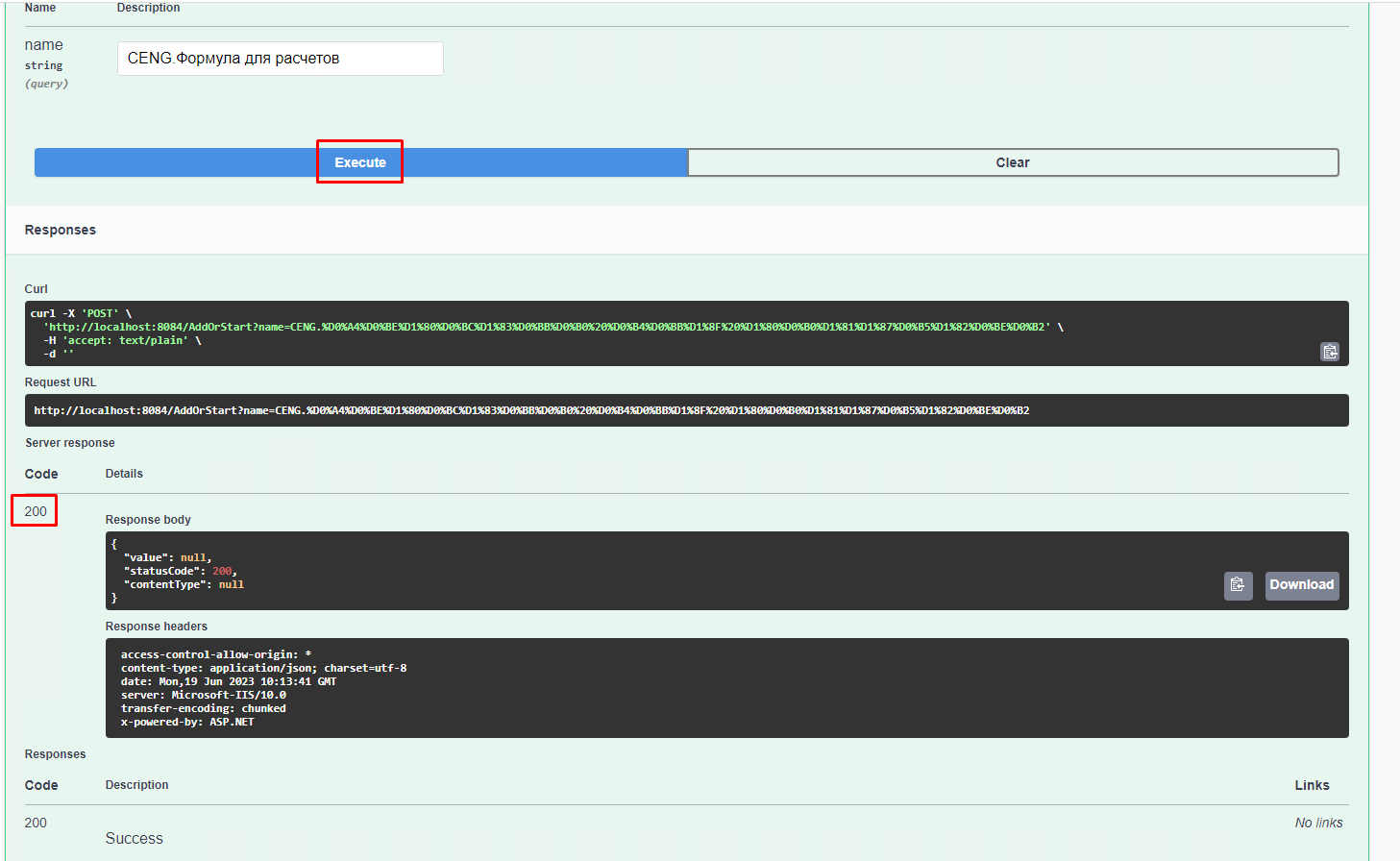
Изображение выглядит как текст, число, линия, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Успешное добавления/ перезапуска узла выдаст код ответа 200:



А также появится соответствующая запись к логе WebApi:



В случае ошибки код ответа будет 400 с соответствующим сообщением об ошибке:

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание



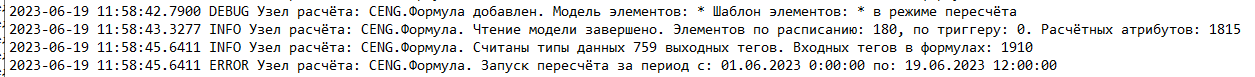
## 2.3 Пересчёт узла расчётов

Выбрать контроллер RecalcNode, нажать Try it out, в поле Name необходимо записать имя атрибута узла расчётов из файла конфигурации Nodes.json приложения, в поля startTimeLocal и entTimeLocal необходимо вписать строкой метки времени начала и окончания пересчёта соответственно в местном часовом поясе. Успешный запуск пересчёта выдаст код ответа 200:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

При старте и окончании пересчёта в логе WebApi будет сделана соответствующая запись:



## 2.4 Остановка узла расчётов

Выбрать контроллер StopNode, нажать Try it out, в поле Name необходимо записать имя атрибута узла расчётов из файла конфигурации Nodes.json приложения. Успешная остановка узла выдаст код ответа 200 с соответствующей записью в лог:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

