УДК 004

Обращение к данным метеостанций на yii2 с помощью REST API

Вихляев Дмитрий Романович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье рассматривается способ обращения к данным метеостанций на сайте yii2 с помощью подключённого REST API. Для реализации будут использоваться готовые данные метеостанций в формате «csv». В результате исследования будет приведён пример обращения к данным из базы данных с помощью http запросов с сайта построенном на yii2.
Ключевые слова: yii2, csv, REST API.

Accessing weather station data on yii2 using the REST API

Vikhlyaev Dmitry Romanovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article discusses how to access weather station data on the yii2 website using the connected REST API. Ready-made weather station data in "csv" format will be used for implementation. As a result of the study, an example of accessing data from a database using http requests from a site built on yii2 will be given.
Keywords: yii2, csv, REST API.

1 Введение

1.1 Актуальность

В настоящее время, веб-приложения вышли за пределы обычных браузеров. Все чаще используются приложения, разработанные с учетом спецификации конкретного устройства. В итоге для одного приложения может быть десяток типов клиентских приложений. Но суть самого приложения не меняется. REST API – это способ сделать универсальное бекенд-приложение, которое не придется переделывать при добавлении нового типа клиента. REST – это архитектурный стиль взаимодействия компонентов распределенного приложения в сети интернет. Его используют для обеспечения взаимодействия между различными клиентскими приложениями с единым сервером. Основные преимущества использования REST заключаются в высокой производительности и надежности. В большинстве случаев взаимодействие происходит с помощью протокола HTTP документами в формате JSON или XML. Для взаимодействия используются различные методы HTTP-запроса, такие как GET, POST,

DELETE, OPTIONS используемые для получения, отправки, удаления данных и описания параметров соединения с ресурсом соответственно. Существует множество различных инструментов для разработки REST API. Одним из самых популярных является использование php-фреймворка yii2.

1.2Обзор исследований

Д.С.Жаворонков, А.А.Бабкина, Е.Ю.Довгий рассмотрели достоинства фреймворка yii2 в разработке rest систем [1]. Ю.М.Закирова разработала rest api сервис на базе php фреймворка yii2 [2]. Ю.Р.Акинин, А.В.Барабанов, Н.И.Гребенникова создали rest api сервиса на основе облачных технологий azure [3]. Д.А.Константинов провёл исследование разработки rest сервиса службы технической поддержки с помощью ASP.NET WEB API [4]. П.А.Безрук изучил системы распределенного мониторинга компьютерной сети на основе rest api [5].

1.3Цель исследования

Цель исследования – используя данные метеостанций загруженные в базу данных создать сайт на yii2 способный обращаться к данным при помощи http запросов.

2 Материалы и методы

Для реализации используются готовые данные метеостанций в формате «csv», php-фреймворк yii2 с подключением REST API.

3 Результаты и обсуждения

Изначальные данные метеостанций хранятся в формате «csv» (рис. 1).

STATION	DATE	SOURCE	LATITUDE	LONGITUDE	ELEVATION	NAME	REPORT_TYPE	CALL_SIGN	QUALITY_CONTROL	WND	CIG	VIS
31713099999	2000-01-01T00:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	050000,1
31713099999	2000-01-01T03:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	050000,1
31713099999	2000-01-01T06:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	050000,1
31713099999	2000-01-01T09:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	999999,9
31713099999	2000-01-01T12:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	999999,9
31713099999	2000-01-01T15:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	020,1,N,0020,1	22000,1,9,N	999999,9
31713099999	2000-01-01T18:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	999999,9
31713099999	2000-01-01T21:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	22000,1,9,N	999999,9
31713099999	2000-01-02T00:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	050000,1
31713099999	2000-01-02T03:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	360,1,N,0020,1	99999,9,9,N	050000,1
31713099999	2000-01-02T06:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	050,1,N,0020,1	99999,9,9,N	010000,1
31713099999	2000-01-02T09:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	999999,9
31713099999	2000-01-02T12:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	999999,9
31713099999	2000-01-02T15:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	999999,9
31713099999	2000-01-02T18:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	999999,9
31713099999	2000-01-02T21:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	340,1,N,0020,1	99999,9,9,N	999999,9
31713099999	2000-01-03T00:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	090,1,N,0020,1	99999,9,9,N	002000,1
31713099999	2000-01-03T03:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	090,1,N,0020,1	99999,9,9,N	002000,1
31713099999	2000-01-03T06:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	090,1,N,0020,1	99999,9,9,N	002000,1
31713099999	2000-01-03T09:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	090,1,N,0020,1	99999,9,9,N	999999,9
31713099999	2000-01-03T12:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	090,1,N,0040,1	99999,9,9,N	999999,9
31713099999	2000-01-03T15:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	999,9,C,0000,1	99999,9,9,N	999999,9
31713099999	2000-01-03T18:00:00	4	48.7333333	132.95	80.0	BIROBIDZHAN, RS	FM-12	99999	V020	340,1,N,0020,1	99999,9,9,N	999999,9

Рис. 1. Данные в формате «csv»

Чтобы экспортировать данные в базу данных mysql, можно использовать запрос на загрузку данных файла в таблицу. Перед этим в базе данных нужно создать таблицу с соответствующими полями и типами данных. За загрузку данных с фала отвечает запрос «LOAD DATA INFILE» в качестве параметра указывается путь к файлу. Далее необходимо указать таблицу, в которую будут заноситься данные, указав атрибут «INTO

TABLE». Данные хранящиеся в формате «csv», по умолчанию разделены запятой и ограниченны кавычками. Для разделения каждого поля при переносе в базу данных надо указать знак разделения при помощи «FIELDS TERMINATED BY», а также начало и конец, используя «ENCLOSED BY». Каждая строка «csv» файла завершается символом новой строки, обозначающим «TERMINATED BY» с параметром «\n». В имеющемся файле первая строка содержит записи названия столбцов, её в базу данных загружать не нужно, поэтому используется «IGNORE» в котором указана первая строка (рис. 2).

```
LOAD DATA INFILE 'c:/tmp/meteodata.csv'
INTO TABLE meteodata
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

Рис. 2. Экспортирование данных из файла «csv» в базу данных

Чтобы взаимодействовать с таблицей из базы данных, в yii2 необходимо создать модель. Для этого в папке models нужно создать файл с названием соответствующим названию таблицы в расширении php. В нём создаётся класс подключаемой таблицы, унаследованный от класса «ActiveRecord» (рис. 3).

```
<?php
namespace app\models;
use yii\db\ActiveRecord;

class Metiostation extends ActiveRecord
{
}
?>
```

Рис. 3. Создание модели таблицы в yii2

Далее в папке контроллер создаётся контроллер, в котором описан одноимённый класс, унаследованный от rest\ActiveController. Внутри класса создаётся публичный объект, которому присваивается содержимое модели таблицы, тем самым указав контроллеру, к какой модели ему необходимо обращаться для редактирования или выборки данных (рис. 4).

```
namespace app\controllers;
use app\models\Metiostation;
use yii\rest\ActiveController;

class MetiostationController extends ActiveController
{
    public $modelClass = 'app\models\Metiostation';
```

Рис. 4. Создание контроллера подключаемого к модели

В yii2 подключение к REST API требует настройки правил url. Чтобы их настроить надо в файле config/web.php изменить компонент «urlManager». Добавление правил контроллеру, которые предоставляют доступ к данным через логические http глаголы, используя класс «rest\UrlRule». Свойство «pluralize» учитывает множественную форму при запросе данных и по умолчанию является «true». Если оставить его без изменений, то при выводе множества данных необходимо будет добавлять множественное окончание к названию контроллера (рис. 5).

```
'urlManager' => [
    'enablePrettyUrl' => true,
    'enableStrictParsing' => true,
    'showScriptName' => false,
    'rules' => [
        [
            'class' => 'yii\rest\UrlRule',
            'controller' => 'metiostation',
            'pluralize'=> false,
            ]
```

Рис. 5. Настройка правил url для подключения к REST API

При правильных настройках при заходе на сайт должен выводиться «xml» формат структуры таблицы из базы данных. Все данные находятся в теге «response», каждая запись внесена в теге «item». К каждой записи можно обращаться по id, добавив к url сайта «/id» (рис. 6).

```
▼<response>
  ▼<item>
     <id>1</id>
     <id_station>31713099999</id_station>
   </item>
  ▼<item>
     <id>2</id>
     <id_station>31725099999</id_station>
   </item>
  ▼<item>
     <id>3</id>
     <id_station>31702099999</id_station>
   </item>
  ▼<item>
     <id>4</id>
     <id_station>31707099999</id_station>
   </item>
 </response>
```

Рис. 6. Вывод данных в «xml» формате

В Yii2 можно использовать не только «xml» формат, но и «json». Для того чтобы API мог принимать данные в формате «json», нужно в файле config/web.php сконфигурировать «parsers» свойство у компонента «request» application component на использование yii\web\JsonParser «json» данных на входе (рис. 7).

```
'components' => [
    'request' => [
        // !!! insert a secret key in the following (if it is empty)
        'cookieValidationKey' => 'i1NR24FoDJ6Hju15DKTwPqs1c_jYq3Uh',
        'parsers' => [
            'application/json' => 'yii\web\JsonParser',
        ],
```

Рис. 7. Настройка свойства для использования формата «json»

Внутри контроллера создаётся функция регулирующая поведение отображаемых данных. В ней указывается, через какую переменную будет указываться формат, а также сами форматы. По умолчанию данные будут выводиться в формате «xml», но при введении «GET» параметра «_format» формат изменится на «json» при присвоении данному параметру значения «json» (рис. 8,9).


```
[{"id":1,"id_station":"31713099999"},
{"id":2,"id_station":"31725099999"},
{"id":3,"id_station":"31702099999"},
{"id":4,"id_station":"31707099999"}]
```


Рис. 8. Вывод данных в «json» формате

```
public function behaviors()
{
    return [
        'contentNegotiator' => [
            'class'=>\yii\filters\ContentNegotiator::class,
            'formatParam'=>'_format',
            'formats' =>[
                'xml'=>\yii\web\Response::FORMAT_XML,
                'application/json'=>\yii\web\Response::FORMAT_JSON
```

Рис. 9. Функция, определяющая в каком формате, будут выводиться данные

В качестве основной таблицы для вывода данных используется таблица, хранящая номера метеостанций. Данные самих станций будут выводиться, только когда пользователь запросит определённый город, к которому относится метеостанция, при помощи связей между таблицами.

Чтобы создать связь между таблицами в yii2 нужно создать новую модель таблицы, к которой будет связана старая. Для этого создаётся новый метод в модели, к которой нужно привязать таблицу. Данный метод возвращает связь с классом модели, в которой указывается какому полю связываемой таблицы, соответствует поле данной таблицы. За реализацию связи один ко многим отвечает метод «hasMany» (рис. 10).

```
public function getBirobidzhan(){
    return $this->hasMany(Birobidzhan::class,['STATION'=>'id_station']);
}
public function getObluche(){
    return $this->hasMany(Obluche::class,['STATION'=>'id_station']);
}
public function getSmidovich(){
    return $this->hasMany(Smidovich::class,['STATION'=>'id_station']);
}
public function getEcaterino_nicolskoe(){
    return $this->hasMany(Ecaterino_nicolskoe::class,['STATION'=>'id_station']);
}
```

Рис. 10. Создание связей между таблицами

В каждой модели можно указать, какие данные включать в представление ресурса в виде массива путём переопределения методов «fields()» и «extraFields()». Методе «fields()» определяет набор полей,

которые всегда будут включены в массив. Без данного метода будут выводиться абсолютно все поля из таблицы. При формировании «GET» запроса, параметру «fields» указываются поля, которые необходимо вывести. Если исключить из запроса данный параметр то выведутся все поля, которые указаны в методе «fields()» (рис. 11).

```php
public function fields()
{
    return [
        'DATE_AND_TIME',
        'TMP',
        'DEW',
        'AA1'
    ];
}
```

Рис. 11. Данные включающиеся в представление ресурса

Метод «extraFields()» определяет дополнительные поля, которые пользователь может запросить через «GET» параметр «expand». В данном методе возвращается список связанных таблиц. Если в запросе не указать данный параметр, то выведутся данные из всех таблиц (рис. 12,13).

```php
public function extraFields()
{
    return [
        'birobidzhan',
        'obluche',
        'smidovich',
        'ecaterino_nicolskoe'
    ];
}
```

Рис. 12. Данные выводимые только при запросе

```
<response>
  <item>
    <id>1</id>
    <id_station>31713099999</id_station>
    <birobidzhan>
      <item>
        <DATE_AND_TIME>2021-01-01T00:00:00</DATE_AND_TIME>
        <TMP>-0310,1</TMP>
        <DEW>-0344,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T03:00:00</DATE_AND_TIME>
        <TMP>-0228,1</TMP>
        <DEW>-0291,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T06:00:00</DATE_AND_TIME>
        <TMP>-0191,1</TMP>
        <DEW>-0306,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T09:00:00</DATE_AND_TIME>
        <TMP>-0259,1</TMP>
        <DEW>-0321,1</DEW>
        <AA1>12,0000,9,1</AA1>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T12:00:00</DATE_AND_TIME>
        <TMP>-0278,1</TMP>
        <DEW>-0325,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T15:00:00</DATE_AND_TIME>
        <TMP>-0309,1</TMP>
        <DEW>-0342,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T18:00:00</DATE_AND_TIME>
        <TMP>-0315,1</TMP>
        <DEW>-0351,1</DEW>
        <AA1/>
      </item>
      <item>
        <DATE_AND_TIME>2021-01-01T21:00:00</DATE_AND_TIME>
        <TMP>-0317,1</TMP>
        <DEW>-0351,1</DEW>
        <AA1>12,0000,9,1</AA1>
      </item>
```


Рис. 13. Вывод связанных данных

Таким несложным способом на сайт yii2 был подключен «REST API», с возможностью взаимодействовать с таблицами данных метеостанций через http запросы.

Библиографический список

1. Жаворонков Д.С., Бабкина А.А., Довгий Е.Ю. Достоинства фреймворка yii2 в разработке rest систем// В сборнике: "Чистая наука" на службе научно-технического прогресса. Сборник статей по итогам Международной научно-практической конференции. 2018. С. 9-11.
2. Закирова Ю.М. Разработка rest api сервиса на базе php фреймворка yii2// В сборнике: Научные исследования в современном мире: опыт, проблемы и перспективы развития. сборник научных статей по материалам III Международной научно-практической конференции. Уфа, 2020. С. 36-40.
3. Акинин Ю.Р., Барабанов А.В., Гребенникова Н.И. Быстрое создание rest api сервиса на основе облачных технологий azure// Вестник Воронежского государственного технического университета. 2012. Т. 8. № 12-1. С. 66-68.
4. Константинов Д.А. Разработка rest сервиса службы технической поддержки с помощью ASP.NET WEB API// В сборнике: Новые информационные технологии в научных исследованиях и в образовании "НИТ 2014". материалы XIX Всероссийской научно-технической конференции студентов, молодых ученых и специалистов. 2014. С. 140-141.

5. Безрук П.А. Разработка системы распределенного мониторинга компьютерной сети на основе rest api// Актуальные проблемы авиации и космонавтики. 2017. Т. 2. № 13. С. 94-95.
6. Зыков А.А. Разработка rest web-api на основе java spring framework// В сборнике: XX Всероссийская студенческая научно-практическая конференция Нижневартовского государственного университета. сборник статей. Ответственный редактор А.В. Коричко. 2018. С. 317-321.
7. Зотова Ю.А., Котилевец И.Д. Разработка архитектуры rest api для взаимодействия с сервисами приложения// В сборнике: Информационные технологии и математическое моделирование систем 2018. труды международной научно-технической конференции. 2018. С. 61-65.
8. Пастушенко В.А. Разработка rest api на flask// Постулат. 2019. № 7 (45). С. 6.
9. Ильясова Ф.С. Стек технологий для создания rest api системы// Наука и бизнес: пути развития. 2019. № 8 (98). С. 66-68.