Рисунок 11.5.3

11.6. HTTP-запросы

11.6.1. Работа со списками через API Web-клиента

Для работы с описанными ниже запросами по пользовательским спискам, необходимо, чтобы версия Web-клиента была не ниже 2.14

HTTP запросы позволяют использовать возможности Web-клиента Автомаршал в сторонних приложениях. С помощью HTTP запросов возможно реализовать получение данных из Web-клиента Автомаршал и передачу запросов на добавление, изменение и удаление записей в пользовательских списках.

Авторизация

Перед началом работы необходимо пройти авторизацию, отправив следующий запрос:

```
POST /login
```

POST запрос на авторизацию должен отправлять на сервер сериализованный в JSON объект с следующими полями:

- username - string; логин пользователя.
- password - string; пароль пользователя.
- isRememberMe - boolean; необходимость возобновления сессии после перезапуска браузера.

Пример сериализованного JSON объекта, отправляемого на сервер для авторизации под учётной записью гостя:

```
{
    username:    "guest",        password:    "",        isRememberMe: false,
}
```

Пример ответа в формате JSON:

```
{
    "redirectUrl": "/",
    "isAutorized": true
}
```

Пример ответа в формате XML:

```
   <LoginResult xmlns="http://schemas.datacontract.org/2004/07/
Automarshal.Http.Framework.Entities.Login" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
       <IsAutorized>true</IsAutorized>
       <RedirectUrl>/</RedirectUrl>
   </LoginResult>
```

Помимо данных (JSON/XML), сервер вернёт cookie-переменную s.

Для дальнейшей работы необходимо передавать cookie-переменную s на сервер с каждым запросом. В противном случае, сервер будет считать, что запрос пришел от не авторизованного пользователя.

Запросы

1. **Получение массива списков**

   Получить все списки с целью выявления идентификатора нужного списка.

   HTTP GET: http://localhost:45555/api/v1/vehiclelists?offset=0&count=20

   Параметры:

   offset - смещение;

   count - необходимое количество (максимум 35).

   В ответ получим следующий объект, состоящий из двух полей entries и _metadata:

   1. entries - массив полученных списков.

   Сущность списка состоит из следующих полей:

   id - long; возвращает и устанавливает идентификатор списка;

   displayName - string; возвращает отображаемое имя списка; *name* -

   string; возвращает имя списка; *color* - string; возвращает и

   устанавливает цвет списка; *order* - int; возвращает и устанавливает

   порядковый приоритет списка; *status* - int; возвращает и

   устанавливает статус списка; *fields* - VehicleListFieldEntry; массив

   [полей списка],

   - id – int; возвращает и устанавливает идентификатор поля списка;
   - displayName – string; возвращает название поля списка.

   passTemplateId – long; возвращает и устанавливает идентификатор шаблона пропуска;

   passTemplate – PassTemplateEntry; сущность шаблона пропуска,

   - id – long; возвращает и устанавливает уникальный идентификатор записи;
   - totalCount – int; возвращает и устанавливает максимально возможное количество проездов;

- displayName – string; объект, содержащий отображаемое имя пропуска;
- allowedPeriod – int; разрешенный период в течении дня (10-15);
- schedules – ScheduleEntry; массив расписаний пропуска,

  Сущность расписания пропуска (допускается только одна сущность в массиве):

- id – long; возвращает и устанавливает уникальный идентификатор записи;
- beginTime – string; возвращает и устанавливает дату начала временного отрезка работы (UTC);
- endTime – string; возвращает и устанавливает дату окончания временного отрезка работы (UTC);
- beginTimeOfDay – int; возвращает и устанавливает начало временного отрезка работы (в мс от начала суток);
- endTimeOfDay – int; возвращает и устанавливает окончание временного отрезка работы (в мс от начала суток);
- Возвращает и устанавливает значение, указывающее, выбран ли временной отрезок работы в указанный день:

  mon – boolean;

  tue – boolean;

  wed – boolean;

  thu – boolean; fri

  – boolean; sat –

  boolean; sun –

  boolean.

- workDay – boolean; возвращает и устанавливает значение, указывающее, выбран ли будний день рабочего календаря;
- dayOff – boolean; возвращает и устанавливает значение, указывающее, выбран ли выходной день рабочего календаря.

defaultVehicleTypeId – long; возвращает идентификатор типа ТС по умолчанию;

defaultVehicleType – VehicleType; возвращает тип ТС по умолчанию,

Сущность типа ТС по умолчанию:

- comment – string; комментарий к типу ТС, например: «Тип ТС по умолчанию»;
- description – string; описание к типу ТС, например: «Тип ТС был создан автоматически системой»;
- id – long; идентификатор типа ТС;

- isDefault – boolean; Возвращает и задаёт тип ТС по умолчанию;
- name – string; название типа ТС;
- spaceRatio – double; возвращает и задаёт значение занимаемого места типа ТС.

constraints – ListsConstraints; возвращает ограничения списка.

2. **_metadata***offset* – int; текущее смещение; *limit* – int; кол-во полученных списков; *totalCount* – int; сколько списков всего в БД.

Пример ответа приведён ниже:

```
{
 "entries": [
   {
     "id": 2,
     "displayName": "Guest passes",
     "name": "guestPassesList",
     "color": "#33CC33",
     "order": 0,
     "status": 0,
     "fields": [
       {
         "id": 5,
         "displayName": "Record made by"
       }
     ],
     "passTemplateId": 0,
```

```
      "passTemplate": {
        "id": 2,
        "totalCount": 2147483647,
        "displayName": "",
        "allowedPeriod": null,
        "schedules": [
          {
            "id": 5,
            "beginTime": null,
            "endTime": null,
            "beginTimeOfDay": null,
            "endTimeOfDay": null,
            "mon": true,
            "tue": true,
            "wed": true,
            "thu": true,
            "fri": true,
            "sat": true,
            "sun": true,
            "workDay": true,
            "dayOff": true
          }
        ]
      },
      "defaultVehicleTypeId": 1,
"defaultVehicleType":
      {
        "comment": "Тип ТС по умолчанию"
        "description": "Тип ТС был создан автоматически системой"
        "id": 1
        "isDefault": true
        "name": "Неизвестный тип ТС"
        "spaceRatio": 1
      },
      "constraints": null
    }
  ],
  "_metadata": {
    "offset": 0,
    "limit": 20,
    "totalCount": 27
  }
}
```


2. **Получение типов ТС**

Для получения возможности выбора типа ТС добавляемой записи.

HTTP GET: http://localhost:45555/api/v1/vehicletypes

В ответ получим следующий массив объектов:


```
[
  {
    "id": 5,
    "name": "Микроавтобусы",
    "description": "",
    "comment": "",
    "spaceRatio": 1.5,
    "isDefault": false
  },
  {
    "id": 1,
    "name": "Неизвестный тип ТС",
```

```
    "description": "Тип ТС был создан автоматически системой",
    "comment": "Тип ТС по умолчанию",
    "spaceRatio": 1.0,
    "isDefault": true
  }
]
```

3. **Получение записей списка**

HTTP GET: http://localhost:45555/api/v1/vehiclelist/records?id=95&offset=0&count=20&searchquery=

Параметры запроса: id - идентификатор списка; offset - смещение;

count - количество записей которое нужно получить (максимум

35); searchquery - запрос для фильтрации записей (пример:

Иванов).

1. entries – массив полученных записей *id* – long; возвращает и

устанавливает идентификатор записи; *plate* – string; возвращает и

устанавливает номер ТС; *fieldValues* – VehicleListRecordFieldValue;

возвращает значения полей,

- id – long; возвращает идентификатор;
- fieldId – long; возвращает идентификатор поля, к которому относится значение;
- value – string; возвращает значение поля.

passes – PassEntry; возвращает и устанавливает список пропусков,

- id – long; возвращает и устанавливает уникальный идентификатор записи;
- totalCount – int; возвращает и устанавливает максимально возможное количество проездов;
- currentCount – int; возвращает и устанавливает количество проездов;
- flags – int; возвращает и устанавливает текущее состояние пропуска;
- createdOn – DateTime; возвращает и устанавливает дату/время создания пропуска

  (UTC); • comment – string; возвращает и устанавливает комментарий;
- recordId – long; возвращает и устанавливает уникальный идентификатор связанной записи в пользовательской таблице;
- createdById – long; возвращает и устанавливает идентификатор пользователя, во время работы которого была создана запись;
- schedules – ScheduleEntry; возвращает и устанавливает временные отрезки работы (см. schedules).

vehicleTypeId – long; возвращает и устанавливает идентификатор типа ТС;

vehicleType – VehicleType; возвращает и устанавливает тип ТС.

2. _metadata

(см. metadata)

В ответ получим следующий объект:


```
{
  "entries": [
    {
      "id": 170,
      "plate": "A222AA22",
      "fieldValues": [
        {
          "id": 152,
          "fieldId": 39,
          "value": "User2"
        }
      ],
      "passes": [
        {
          "id": 165,
          "totalCount": 2147483647,
          "currentCount": 0,
          "flags": 0,
          "createdOn": "0001-01-01T00:00:00",
          "comment": "",
          "recordId": 0,
          "createdById": 0,
          "schedules": [
            {
              "id": 10190,
              "beginTime": "2018-10-31T12:19:55.0000000",
              "endTime": "2018-11-01T12:19:55.0000000",
              "beginTimeOfDay": 28800000,
              "endTimeOfDay": 36000000,
              "mon": true,
              "tue": true,
              "wed": true,
              "thu": true,
              "fri": true,
              "sat": true,
              "sun": true,
              "workDay": true,
              "dayOff": true
            }
          ]
        }
      ],
      "vehicleTypeId": null,
      "vehicleType": null
    },
  ],
  "_metadata": {
    "offset": 0,
    "limit": 13,
    "totalCount": 4
  }
}
```

Пример объекта типа ТС:


```
{
  "comment": ""
  "description": ""
  "id": 3
  "isDefault": false
  "name": "Грузовые"
  "spaceRatio": 1.2
}
```


4. **Удаление**

Имея идентификатор записи списка, можно её удалить:

HTTP DELETE: http://localhost:45555/api/v1/vehiclelist/record?id=260

5. **Добавление записи**

HTTP POST: http://localhost:45555/api/v1/vehiclelist/record?id=100

Параметры запроса:

id - идентификатор списка, в который добавляется запись.

В теле запроса необходимо отправить следующий объект:


```
{
  "Id": 0,
  "Plate": "Test1",
  "VehicleTypeId": 4,
  "VehicleType": {
    "Id": 4,
    "Name": "Автобусы",
    "Description": "",
    "Comment": "",
    "SpaceRatio": 1.4,
    "IsDefault": false
  },
  "FieldValues": [
    {
      "FieldId": 39,
      "Value": "igor"
    }
  ],
  "Passes": [
    {
      "Id": 22,
      "TotalCount": 2147483647,
      "Schedules": [
        {
          "Id": 187,
          "BeginTime": "31.10.2018 12:19:55",
          "EndTime": "01.11.2018 12:19:55",
          "BeginTimeOfDay": 28800000,
          "EndTimeOfDay": 36000000,
          "WorkDay": true,
          "DayOff": true,
          "Mon": true,
          "Tue": true,
          "Wed": true,
          "Thu": true,
          "Fri": true,
          "Sat": true,
```

```
          "Sun": true
        }
      ],
      "AllowedPeriod": null
    }
  ]
}
```

Создана следующая

запись: Номер -

Test1 id типа ТС - 4

Тип ТС - Автобусы

Поле списка (39) содержит запись igor

Пропуск не ограничен по количество проездов

Пропуск действителен с 31.10.2018 12:19:55 по 01.11.2018 12:19:55

Пропуск действителен с 8-10 утра

Пропуск действует в любой день недели

6. **Изменение записи**

Имея идентификатор записи списка, можно её изменить:

HTTP PUT: http://localhost:45555/api/v1/guestslist/record

В теле запроса необходимо отправить изменённую запись.

ВАЖНО:

Идентификаторы изменённой и изменяемой записи должны совпадать (кроме типа ТС, он может быть изменён). Имеются ввиду такие идентификаторы как: id записи, id пропуска, id расписания, идентификаторы полей списка и полей в изменённой записи (массив объектов FieldValues).