Инструкция

Описание API

Mirapolis 4.xx

Версия инструкции 2.41

Mirapolis

Содержание

Настройка системы для работы с API ... 8
Идентификатор приложения и секретный ключ ... 8
Создатель объектов ... 8
Синтаксис вызова ... 9
Постраничное возвращение результата ... 11
Стандартные методы ... 12
Получение полного списка полей модуля ... 12
Запрос возвращения значений дополнительных полей ... 12
Изменение дополнительных полей ... 12
Получение (поиск) списка объектов ... 12
Расширенная фильтрация списка объектов ... 13
Получение (поиск) идентификаторов объектов ... 14
Формат параметров ... 15
Дата ... 15
Строка ... 15
Число ... 15
Малое число ... 15
Логическое ... 15
Пустое значение ... 15
Файл ... 15
Функции доступные для вызова ... 17
Модуль auth (Авторизация) ... 17
Открытие пользовательской сессии ... 17
Закрытие пользовательской сессии ... 18
Продление пользовательской сессии ... 18
Модуль persons (Физические лица) ... 18
Добавление физического лица ... 19
Получение информации о физическом лице ... 19
Получение информации о физическом лице по логину ... 19
Изменение информации о физическом лице ... 20
Удаление физического лица ... 20
Получение списка сессий входа в систему ... 20
Получение последней сессии входа в систему ... 20
Модуль personGroups (Группы физических лиц) ... 21
Создание группы физических лиц ... 21
Получение информации о группе физических лиц ... 21
Удаление группы физических лиц ... 21
Изменение группы физических лиц ... 22
Добавление дочерней группы ... 22
Исключение дочерней группы ... 22
Получение списка дочерних групп ... 22
Добавление физического лица в группу ... 23
Исключение физического лица из группы ... 23
Получение списка физических лиц, входящих в группу ... 23
Модуль roles (Доступ) ... 23
Добавление роли ... 24
Получение информации о роли ... 24
Получение информации о роли по системному названию ... 24
Получение информации о ролях пользователя ... 24
Изменение информации о роли ... 25
Удаление роли ... 25
Добавление роли пользователю ... 25
Удаление роли пользователя ... 25

Получение информации об организациях, за которые отвечает пользователь 25
Добавление организации, за которую отвечает пользователь 26
Исключение организации, за которую отвечает пользователь 26
Назначение организации по умолчанию 26
Модуль cas (Организации) 26
Добавление организации 27
Получение информации об организации 27
Изменение информации об организации 27
Удаление организации 27
Модуль caGroups (Группы организаций) 29
Создание группы организаций 29
Получение информации о группе организаций 29
Удаление группы организаций 29
Изменение группы организаций 30
Добавление дочерней группы 30
Исключение дочерней группы 30
Получение списка дочерних групп 30
Добавление организации в группу 31
Исключение организации из группы 31
Получение списка организаций, входящих в группу 31
Модуль measures (Мероприятия) 31
Добавление мероприятия 32
Получение информации о мероприятии 33
Изменение информации о мероприятии 33
Удаление мероприятия 33
Получение информации об участниках мероприятия 34
Добавление участника мероприятия 34
Добавление участника мероприятия по E-mail 34
Исключение участника мероприятия 35
Отправка приглашения пользователю на мероприятие 35
Получение информации о преподавателях мероприятия 36
Добавление преподавателя на мероприятие 36
Удаление преподавателя мероприятия 36
Получение ссылки гостевого входа в вебинар 36
Получение ссылки для входа участника мероприятия в вебинар 37
Получение ссылок на записи вебинара 37
Получение результатов участия в мероприятии 37
Изменение дат обучения слушателя на мероприятии 37
Изменение дат обучения слушателя на мероприятии, зарегистрированного по источнику 38
Получение компонентов программы мероприятия 38
Получение списка ресурсов мероприятия 38
Завершение мероприятия 38
Получение списка вебинар-опросов мероприятия 39
Модуль myMeasures (Мои мероприятия) 39
Получение списка мероприятий-вебинаров участника 40
Получение списка мероприятий пользователя в качестве участника 40
Получение списка мероприятий пользователя в качестве преподавателя 40
Получение списка мероприятий авторизованного пользователя 40
Модуль measureResults (Результаты участия в мероприятии) 42
Добавление результата 43
Получение информации о результате 43
Изменение результата 43
Удаление результата 43
Получение информации о результатах по разделам 44
Модуль measureProgress (Прогресс участия в мероприятии) 44

Добавление прогресса 44
Получение информации о прогрессе 45
Изменение прогресса 45
Удаление прогресса 45
Модуль measureAttempts (Попытки прохождения мероприятия) 45
Добавление попытки 46
Получение информации о попытке 46
Изменение попытки 46
Удаление попытки 47
Модуль measureInteractions (Данные прохождения тестирований) 47
Добавление ответа участника на вопрос 47
Получение информации об ответе участника на вопрос 48
Изменение ответа участника на вопрос 48
Удаление ответа участника на вопрос 48
Модуль measureGroups (Группы мероприятий) 48
Создание группы мероприятий 49
Получение информации о группе мероприятий 49
Удаление группы мероприятий 49
Изменение группы мероприятий 49
Добавление дочерней группы 50
Исключение дочерней группы 50
Получение списка дочерних групп 50
Добавление мероприятия в группу 50
Исключение мероприятия из группы 51
Получение списка мероприятий, входящих в группу 51
Модуль measureUserGroups (Пользовательские группы мероприятий) 51
Добавление пользовательской группы 52
Получение информации о пользовательской группе 52
Изменение информации о пользовательской группе 52
Удаление пользовательской группы 53
Добавление дочерней пользовательской группы 53
Исключение дочерней пользовательской группы 53
Получение списка дочерних пользовательских групп 53
Добавление мероприятия в пользовательскую группу 53
Исключение мероприятия из пользовательской группы 54
Получение списка мероприятий пользовательской группы 54
Модуль surveys (Вебинар-опросы) 54
Добавление вебинар-опроса 55
Получение информации о вебинар-опросе 56
Изменение информации о вебинар-опросе 56
Удаление вебинар-опроса 56
Получение списка ответов на вебинар-опрос 56
Получение статистики по ответам на вебинар-опрос 56
Получение ответов пользователей на вебинар-опрос 57
Получение свободных ответов пользователей на вебинар-опрос 57
Модуль reportTemplates (Шаблоны отчетов) 57
Добавление шаблона отчета 62
Получение информации о шаблоне отчета 63
Изменение информации о шаблоне отчета 63
Удаление шаблона отчета 63
Построение отчета с возвращением данных в JSON 63
Построение отчета с возвращением данных в XML 64
Модуль reports (Отчеты) 65
Добавление отчета 65
Получение информации об отчете 65

Изменение информации об отчете ... 66
Удаление отчета ... 66
Модуль certificates (Выданные сертификаты) ... 66
Получение информации сертификате ... 66
Изменение информации о выданном сертификате ... 66
Удаление выданного сертификата ... 67
Получение списка сертификатов пользователя ... 67
Модуль certTypes (Шаблоны сертификатов) ... 67
Получение списка шаблонов сертификатов по мероприятиям пользователя ... 67
Модуль mediaResources (Ресурсы медиатеки) ... 68
Добавление ресурса ... 69
Получение информации о ресурсе ... 69
Изменение информации о ресурсе ... 69
Удаление ресурса ... 69
Модуль mediaUserGroups (Пользовательские группы ресурсов) ... 70
Добавление пользовательской группы ... 70
Получение информации о пользовательской группе ... 70
Изменение информации о пользовательской группе ... 71
Удаление пользовательской группы ... 71
Добавление дочерней пользовательской группы ... 71
Исключение дочерней пользовательской группы ... 71
Получение списка дочерних пользовательских групп ... 72
Добавление ресурса в пользовательскую группу ... 72
Исключение ресурса из пользовательской группы ... 72
Получение списка ресурсов пользовательской группы ... 72
Модуль mediaGroups (Группы ресурсов) ... 72
Добавление группы ... 73
Получение информации о группе ... 73
Изменение информации о группе ... 73
Удаление группы ... 73
Добавление дочерней группы ... 74
Исключение дочерней группы ... 74
Получение списка дочерних групп ... 74
Добавление ресурса в группу ... 74
Исключение ресурса из группы ... 74
Получение списка ресурсов группы ... 75
Модуль favorites (Избранное) ... 75
Добавление в избранное ... 75
Получение избранного пользователя ... 75
Получение избранного пользователя по определенному объекту ... 76
Удаление записи избранного ... 76
Модуль tests (Тестирование) ... 76
Получение данных тестирования ... 79
Модуль messages (Личные сообщения) ... 79
Добавление темы ... 80
Получение информации о теме ... 80
Изменение информации о теме ... 80
Удаление темы ... 80
Добавление сообщения ... 81
Получение информации о сообщении ... 81
Изменение сообщения ... 81
Удаление сообщения ... 81
Получение списка сообщений темы ... 82
Модуль comments (Комментарии) ... 82
Добавление комментария ... 82

Получение информации о комментарии 83
Изменение информации о комментарии 83
Удаление комментария 83
Модуль galleries (Галереи) 83
Добавление галереи 83
Получение информации о галерее 84
Изменение информации о галерее 84
Удаление галереи 84
Модуль importTemplates (Шаблоны импорта) 84
Добавление шаблона импорта 84
Получение информации о шаблоне импорта 85
Изменение информации о шаблоне импорта 85
Удаление шаблона импорта 85
Модуль importTasks (Задачи импорта) 85
Добавление задачи импорта 86
Получение информации о задаче импорта 86
Изменение информации о задаче импорта 86
Удаление задачи импорта 86
Добавление правила импорта 87
Получение информации о правиле импорта 87
Изменение информации о правиле импорта 87
Удаление правила импорта 87
Получение списка правил импорта задачи импорта 88
Загрузка данных задачи импорта из файла 88
Запуск задачи импорта данных 88
Модуль myProfile (Мой профиль) 88
Получение информации о текущем пользователе 88
Получение информации о группах текущего пользователя 89
Добавление произвольного объекта для пользователя 89
Получение информации об объектах пользователя 89
Изменение информации об объекте пользователя 90
Модуль Vacancy (Вакансии) 90
Добавление вакансии 90
Получение информации о вакансии 90
Изменение информации о вакансии 91
Удаление вакансии 91
Модуль qua (Компетенции) 91
Добавление компетенции 91
Получение информации о компетенции 92
Изменение информации о компетенции 92
Удаление компетенции 92
Получение списка мероприятий компетенции 92
Модуль personAttainments (Присвоенные достижения) 93
Добавление присвоенного достижения 93
Модуль gameAccountChanges (Изменения игрового счета) 93
Добавление изменения игрового счета 94
Модуль currentGameAccount (Игровой счёт пользователя) 94
Получение игрового счёта пользователя 95
Модуль GroupVacancy (Группы вакансий) 95
Создание группы вакансий 95
Получение информации о группе вакансий 95
Удаление группы вакансий 96
Изменение группы вакансий 96
Добавление дочерней группы 96
Исключение дочерней группы 96

Получение списка дочерних групп ... 97
Добавление вакансии в группу ... 97
Исключение вакансии из группы ... 97
Получение списка вакансий, входящих в группу ... 97
Модуль externalRecords (Внешние записи) ... 98
Получение всех внешних записей ... 98
Добавление внешней записи ... 98
Получение информации о внешней записи ... 98
Изменение внешней записи ... 99
Удаление внешней записи ... 99
Модуль ListConstructor (Динамические списки) ... 99
Получение перечня динамических списков ... 99
Получение содержимого динамического списка ... 100
Приложение А. Примеры использования API на PHP ... 100
Функция для генерации подписи ... 113
Функция для отправки запроса ... 113
Создание мероприятия ... 114
Редактирование мероприятия ... 114
Получение информации о мероприятии ... 114
Удаление мероприятия ... 114
Получение количества мероприятий из заголовка Content-Range ... 114
Конвертация строк в другую кодировку ... 115
Конвертация параметров запроса GET в подходящий для url формат ... 115
История изменений документа ... 116

Настройка системы для работы с API

Для работы с API необходимо получить идентификатор приложения и секретный ключ, а также указать пользователя в глобальных настройках системы, который будет использоваться, как создатель объектов.

Идентификатор приложения и секретный ключ

Для сервиса Mirapolis Virtual Room

Идентификатор приложения и секретный ключ отображается на странице **Ключи API**. Страница доступна в меню **Администрирование -** подпункт **Ключ API**. Эта страница доступна только пользователям с ролью **Администратор**.

Для других сервисов

Для перехода на страницу с ключом и идентификатором приложения необходимо настроить пункт меню, ссылающийся на системную страницу **randomsecretkey**, при этом нужно указать в поле **Дополнительные параметры** пункта меню **id=2**. Страница доступна только для авторизованных пользователей с ролью на основе профиля **Администратор**.

Создатель объектов

При работе через API должен быть задан пользователь, от имени которого будут создаваться объекты в системе. В качестве создателя объектов используется пользователь, связанный с сессией, полученной по API. Если запрос выполняется без авторизации по API, используется пользователь, указанный, как администратор системы. Выбор администратора осуществляется на странице **Настройки** - группа полей **Общие настройки** - поле **Администратор**.

На заметку

- *Поле **Администратор** обязательно для заполнения и не может быть оставлено без значения.*
- *Для сервиса Mirapolis Virtual Room в качестве администратора используется учетная запись владельца системы.*

Синтаксис вызова

Вызов API производится с помощью REST запросов. В качестве адреса для вызова используется URL вида: **[Адрес системы]/service/[версия API]/[название модуля]/[вызываемая функция и параметры в URL-пути]?[параметры запроса].**

Например

Метод: GET

Запрос:
https://test.lmsonline.ru/mira/service/v2/persons/3?pfirstname=test&appid=exampleappid&sign=641BD1259DAEC2BEC5341ADB7EBFAE33

В этом запросе:

- *[Адрес системы] - "https://test.lmsonline.ru/mira";*
- *[версия API] - "v2";*
- *[название модуля] - "persons";*
- *[вызываемая функция и параметры в пути] - "/3".*

В данном случае запрашивается объект с id=3 модуля persons (физические лица). Для большинства запросов вызываемая функция определяется по сочетанию модуля и типа HTTP метода.

Для обращений по API в облачном сервисе используйте адрес, по которому Вы работаете с системой через браузер.

При неправильно введенном адресе системы либо модуля API может возвращаться html страница с сообщением «Ресурс не найден (404)» либо другим содержанием.

Типы HTTP – методов:

- GET – получение данных объекта.
- POST – создание объекта.
- PUT – обновление объекта.
- DELETE – удаление объекта.

В большинстве запросов необходимо указывать подпись запроса и идентификатор запроса в параметрах:

- appid – идентификатор приложения.
- sign – генерированная подпись.

Если запрос не требует указание идентификатора приложения и подписи, то это будет явно указано в описании запроса, такие запросы используют аутентифицируются с помощью сессии пользователя.

Подпись **sign** формируется в виде md5 в верхнем регистре от строки вида:
<**путь**>?параметры_запроса&**appid**=...&**secretkey**=...

В данной строке:

- Путь – адрес с учетом протокола, адреса системы, контекста, версии API, модулем и параметрами в пути.
- Параметры запроса – параметры после url. Указываются для обновления, добавления либо поиска объекта. Для расчёта подписи, параметры запроса, за исключением appid и secretkey, должны быть отсортированы в алфавитом порядке. Параметры appid и secretkey должны быть в конце строки запроса.
- secretkey – ключ системы. Ключ системы используется только для формирования подписи **sign**.

Например

Расчёт md5:

sign=strtoupper(md5(https://test.lmsonline.ru/mira/service/v2/persons/3?pfirstname=test&appid=exampleappid&secretkey=secret)) = 641BD1259DAEC2BEC5341ADB7EBFAE33

В этом случае полный запрос будет выглядеть следующим образом:
https://test.lmsonline.ru/mira/service/v2/persons/3?pfirstname=test&appid=exampleappid&sign=641BD1259DAEC2BEC5341ADB7EBFAE33

Внимание!

- *При работе системы на одном сервере приложений и доступной по нескольким адресам (например, test.lmsonline.ru из интернета и test.lmsonline из интрасети, в качестве адреса для формирования подписи нужно использовать только адрес, сконфигурированный в системе. Отправлять запрос можно на любой адрес.*
- *Ключ системы (secretkey) не передается в составе входных параметров запроса к серверу. Он используется только для формирования подписи **sign**.*

Результат выполнения метода возвращается в виде JSON. Код ответа зависит от результата выполнения метода. Коды ответов:

- 200 – Запрос выполнен успешно. При получении этого статуса тело ответа может не возвращаться.
- 201 – Запись создана (только для POST).
- 304 – Данные не изменились (только для PUT).
- 400 – Некорректный запрос.
- 401 – Неавторизованный доступ.
- 404 – Данные не найдены.
- 403 – Доступ запрещен.
- 500 – Внутренняя ошибка сервера.

Функция может вернуть JSON объект содержащий результат выполнения функции (методы GET, POST, PUT).

Например

В случае выполнении функции services/v2/persons/{id} возвращается результат:


```
{
  "pfirstname": "Admin",
  "plastname": "Admin",
  "rspostidname": "",
  "personemail": "test@mirapolis.ru",
  "personid": "0",
  "pilogin": "admin",
  "caidname": "",
  "caid": "",
  "psurname": "Admin",
  "pipassword": "admin",
  "rspostid": ""
}
```


В случае ошибки при обработке запроса возвращается JSON объект с кодом и текстом ошибки.

Например

```
{
   "errorCode": 500,
   "errorMessage": "appid is null"
}
```


Постраничное возвращение результата

Метод API может вернуть массив JSON объектов (методы GET), в этом случае предусмотрен постраничный вывод результата.
В ответе на такие запросы в заголовках ответа возвращается информация о количестве записей в заголовке. Content-Range: items 50-75/100, что означает, что ответ содержит объекты с 50 по 75 из 100.
Для управления постраничным выводом в URL надо добавить параметры limit и offset. Например, persons?limit=25&offset=50 вернет 25 пользователей начиная с пятидесятого. По умолчанию limit=20, offset=0.

Например

При получении списка ролей физического лица возвращается несколько объектов


```
[{
   "roleid": "2",
   "profileid": "2",
   "rolename": "Слушатель",
   "sysrolename": "student",
   "roleisdefault": "1"
}, {
   "roleid": "3",
   "profileid": "0",
   "rolename": "Администратор",
   "sysrolename": "admin",
   "roleisdefault": "0"
}]
```


На заметку

- *Если количество запрашиваемых записей в поле limit превышает количество записей в системе, то вернутся все записи, вне зависимости от значения поля offset.*
- *Максимальное значение limit равно 200.*

Стандартные методы

Получение полного списка полей модуля

Тип запроса: GET

/service/v2/{module}/info

При использовании API возможно задавать значения всех полей объекта, поддерживаемого в API, а также запрашивать получение значений полей, которые по умолчанию не входят в модель данных объекта в API. В качестве названий полей используются системные названия полей. В интерфейсе системы названия полей и тип поля возможно просмотреть в конструкторе представлений. Метод позволяет получить полный список полей объекта, которые можно использовать по API. Получать список полей и запрашивать значения можно только для модуля, для вложенных объектов получать список полей и запрашивать значения нельзя.

Например

https://test.lmsonline.ru/mira/service/v2/measures/info - допустимый запрос.

https://test.lmsonline.ru/mira/service/v2/measures/{measureId}/members/info - недопустимый запрос.

Выходные данные:

- Массив значений «Системное название: значение», где **системное название** – системное название поля, которое возможно использовать в API, а **значение** – название поля.

Запрос возвращения значений дополнительных полей

В методах получения информации об объектах возможно запросить получение значений дополнительных полей, добавив параметр bean_add_fields= с перечислением через запятую системных названий запрашиваемых полей.

/service/v2/{module}/{method}/{param}?bean_add_fields={param1,param2,…}

Например

https://test.lmsonline.ru/mira/service/v2/persons/0?bean_add_fields=categoryid,daid&appid=system&sign=F8CEAC43BCB0AEA4BBFC3B281064E3F8

В выходных параметрах объектов помимо стандартной модели данных будут возвращены значения запрошенных полей.

Изменение дополнительных полей

В методах изменения информации об объектах или создания объектов возможно использовать системные названия полей для изменения значений этих полей наравне с полями модели данных объекта в API.

Получение (поиск) списка объектов

Тип запроса: GET

/service/v2/{module}

Метод позволяет постранично получать список объектов с параметрами модели объекта. Метод возможно использовать в модулях, работающими с объектами системы. Возможно задать параметры искомого объекта для поиска (фильтрации) возвращаемых записей.

Входные параметры:

- Набор параметров модели объекта, по которым нужно осуществить поиск.

Выходные данные:

- Массив объектов с параметрами модели объекта. Данные возвращаются постранично.

Например

https://test.lmsonline.ru/mira/service/v2/persons?caid=&directorid=3&appid=system&sign=9DE5A1E1BEBC82F244CF26608867B96F

*По запросу будет осуществлен поиск физических лиц, не привязанных к организации, с руководителем с id=3 (идентификатор физического лица, указанный в поле **Руководитель**). В ответе будет возвращен массив объектов с параметрами модели физического лица. Поскольку limit и offset не указаны, возвращается не более 20 объектов.*

Расширенная фильтрация списка объектов

Тип запроса: GET

/service/v2/{module}

Расширенная фильтрация позволяет задать произвольный набор фильтров при поиске объектов.

Входные параметры:

- filter – правило фильтрации объектов. Этот параметр может быть указан несколько раз в одном запросе. Правила, указанные в разных параметрах filter, объединяются при фильтрации по логическому условию И.

Выходные данные:

- Массив объектов с параметрами модели объекта. Данные возвращаются постранично.

Формирование правил для параметра filter

- В одном правиле filter можно указать несколько условий фильтрации.
- Несколько условий фильтрации в одном правиле разделяются символом запятой: filter={УсловиеФильтрации1},{УсловиеФильтрацииN}
- Несколько условий фильтрации в одном правиле filter объединяются при фильтрации по логическому условию ИЛИ.
- Каждое условие фильтрации представляет собой строку, формируемую по следующему правилу: {ИмяПараметра}{УсловиеРавенства}{Значение}
  Здесь:
  - {ИмяПараметра} – имя параметра из модели данных искомого объекта
  - {Значение} – искомое значение объекта
  - {УсловиеРавенства} задает условие сопоставления значения параметра искомого объекта со значением для фильтрации. Возможны следующие условия:
    - == (строгое равенство) – условию фильтрации должны соответствовать объекты, для которых значение параметра равно указанному значению.

- =@ (вхождение подстроки в строку) – условию фильтрации должны соответствовать объекты, для которых указанное значение входит в значение параметра как подстрока.

- При формировании правила filter необходимо экранировать символы запятой «,» в строковых параметрах символом «\».
- Сформированное значение параметра filter необходимо кодировать с помощью urlencode аналогично остальным параметрам запроса.

Например

Текстовое представление запроса:

https://test.lmsonline.ru/mira/service/v2/cas?filter=castringcode=@a,castringcode=@b&filter=website=@www&appid=system&sign=4E0E8DABFDB94962CE0ADD2C9E9F9E72

Кодированное представление запроса:

https://test.lmsonline.ru/mira/service/v2/cas?filter=castringcode%3D@a,castringcode%3D@b&filter=website%3D@www&appid=system&sign=4E0E8DABFDB94962CE0ADD2C9E9F9E72

По запросу будет осуществлен поиск организаций, у которых строковый код содержит символ "a" или "b", и веб-сайт содержит подстроку "www".

Например

Текстовое представление запроса:

https://test.lmsonline.ru/mira/service/v2/cas?filter=caname=@A\,+B&appid=system&sign=6C14C44E2D3A2867E483E2731596C70E

Кодированное представление запроса:

https://test.lmsonline.ru/mira/service/v2/cas?filter=caname%3D@A%5C%2C+B&appid=system&sign=6C14C44E2D3A2867E483E2731596C70E

По запросу будет осуществлен поиск организаций, у которых имя содержит подстроку "A, B".

Получение (поиск) идентификаторов объектов

Тип запроса: GET

/service/v2/{module}/search/ids

Метод позволяет получать список идентификаторов объектов, удовлетворяющих критериям поиска. Метод возможно использовать в любом модуле, работающим с объектами системы. Необходимо задать параметры искомого объекта для поиска (фильтрации) возвращаемых записей.

Входные параметры:

- Набор параметров модели объекта, по которым нужно осуществить поиск.

Выходные данные:

- Массив идентификаторов объектов.

Например

https://test.lmsonline.ru/mira/service/v2/persons/search/ids?caid=&directorid=3&appid=system&sign=4638C2AD72D957404F1CF5216C9D4C5C

*По запросу будет осуществлен поиск физических лиц, не привязанных к организации, с руководителем с id=3 (идентификатор физического лица, указанный в поле **Руководитель**). В ответе будет возвращен массив идентификаторов.*

Формат параметров

Дата

Даты передаются в формате **yyyy-MM-dd HH:mm:ss.SSS**

Например

2013-01-12 13:00:00.000

Строка

Строки передаются в кодировке UTF-8 в стандартном представлении для соответствующего метода (urlencode для GET). Максимальная длинна строк ограничена для разных параметров. Строки большей длины, чем максимальная, при помещении в базу данных обрезаются до максимально возможной длины для соответствующего поля либо будет возвращена ошибка валидации данных.

Внимание!

В начале и в конце параметров не допускается использование пробелов.

Подпись рассчитывается от строкового представления параметров, а передавать нужно значение, кодированное с помощью urlencode.

Число

Числа приводятся к строке, т.е. 0 приводится к “0”, но не к “”.
Максимальное значение числовых параметров = 2147483647.

Малое число

Малые числа приводятся к строке, т.е. 0 приводится к “0”, но не к “”.
Максимальное значение малых чисел = 32767.
В качестве малых чисел как правило используются предопределенные значения.

Логическое

Формат передачи:

- «0» либо «false» – ложь
- «1» либо «true»– истина

Пустое значение

Значения типа null для строк и чисел не должны передаваться при создании объекта. В запросах на обновление данных возможно использовать пустые значения для установки значения пустым. В запросах на поиск данных возможно использовать пустые значения для поиска записей с пустым полем.

Файл

Загрузка файлов производится в соответствии со стандартом http://www.ietf.org/rfc/rfc1867.txt.
Пример заголовков запроса на добавление файла:

1. Заголовок **content-type** устанавливается в значение **multipart/form-data**.
2. В заголовке **multipart/form-data** указывается название разделителя частей сообщения.
3. В заголовке **content-length** – длинна сообщения.
4. В заголовке **Content-Disposition** указывается:
    a. В параметре **name** – системное название поля в системе Мираполис, в которое добавляется файл («fileid» в примере – это системное название поля Фото для физического лица и системное название поля Файл ресурса медиатеки).
    b. В параметре **filename** – название, с которым файл будет добавлен в систему.
5. В заголовке **Content-Type** – MIME тип файла.

```
HOST: https://<URL>
content-type: multipart/form-data; boundary=----WebKitFormBoundaryeBYeEBhNJuqYSYQo
content-length: 20083

------WebKitFormBoundaryeBYeEBhNJuqYSYQo
Content-Disposition: form-data; name="fileid"; filename="photo.jpg"
Content-Type: image/jpeg

<Тут данные файла>
------WebKitFormBoundaryeBYeEBhNJuqYSYQo--
```

В качестве названия параметра формы с файлом (fileid в примере) указывается системное название поля с файлом.

Функции доступные для вызова

Модуль auth (Авторизация)

Для запросов модуля auth не обязательно использовать подпись запроса (параметр sign).
Вызов запросов возможен из запущенного электронного курса при прохождении пользователем. Открытие пользовательской сессии по API при этом не требуется, в качестве авторизации запроса используется сессия пользователя на сервере доставки.
Для открытия пользовательской сессии нужно отправить с клиентской части курса запрос POST https://systemURL/mirads/service/auth/login, где «systemURL» - доменное имя системы, а «mirads» - контекст сервера доставки системы.

Внимание!

Для обращений по API из электронных курсов в этом запросе не используется версия API (v2).

Информация о сессии пользователя представляется следующей моделью данных:

Название поля	Описание	Тип
access_token	Идентификатор пользовательской сессии. Название параметра **access_token** используется только в методе открытия пользовательской сессии, в других методах используется название параметра **usrsesid**.	Строка
expires_in	Время действия сессии в секундах. Сессия не продляется автоматически при вызове запросов с указанной сессией. Для продления сессии API нужно использовать метод Продление пользовательской сессии во время, когда сессия действует. Продление завершенной сессии невозможно.	Число

Открытие пользовательской сессии

Тип запроса: POST

/service/v2/auth/login

Параметры запроса:

- login – логин пользователя, для которого открывается сессия. Обязательный параметр при отправке запроса из внешней системы. Не требуется, если запрос отправляется с клиентской части курса, запущенного на сервере доставки.
- password – пароль пользователя, для которого открывается сессия. Обязательный параметр при отправке запроса из внешней системы. Не требуется, если запрос отправляется с клиентской части курса, запущенного на сервере доставки.

Выходные данные:

- Параметры из модели данных сессии.

На заметку

- *Для одного пользователя одновременно действует только один id сессии. При получении нового id сессии, пока действует старый, старый id сессии аннулируется. Для продления действия id сессии нужно использовать метод Продление пользовательской сессии.*
- *Полученный id сессии может быть использован в качестве параметра у запросов других модулей, создающих или обновляющих какой-либо объект. При этом в качестве создателя или изменившего будет выступать тот пользователь, которому соответствует пользователь с указанной сессией. Значение идентификатора сессии для запросов должно передаваться в параметре **usrsesid**.*

– *При работе с идентификатором сессии usrsesid нужно использовать полученный идентификатор до времени окончания действия идентификатора либо до окончания процессов, связанных с идентификатором сессии. Не нужно перегенерировать идентификатор сессии на каждый запрос. Перегенерация идентификатора сессии на сервере доставки вызовет завершение работы пользователя с электронным материалом и приведет к закрытию попытки пользователя.*

Закрытие пользовательской сессии

Тип запроса: POST

/service/v2/auth/logout

Входные параметры:

- usrsesid – id открытой сессии, которую необходимо закрыть

Продление пользовательской сессии

Тип запроса: POST

/service/v2/auth/refresh

Входные параметры:

- usrsesid – id открытой сессии, которую необходимо продлить.

На заметку

– *В результате выполнения запроса параметр expires_in принимает начальное значение.*

Модуль persons (Физические лица)

Информация о физическом лице представляется следующей моделью данных:

Название поля	Описание	Тип
personid	Идентификатор физического лица	Число
plastname	Фамилия	Строка (100)
pfirstname	Имя (для физического лица всегда должно быть заполнено)	Строка (100)
psurname	Отчество	Строка (100)
ppsex	Пол (0 – Мужской, 1 – Женский). Если не задан, используется мужской.	Малое
isuser	Является ли физическое лицо пользователем (false – нет, true -да)	Логическое
pilogin	Логин	Строка (150)
pipassword	Пароль	Строка (150)
caid	Идентификатор организации	Число
caidname	Название организации	Строка (255)
rspostid	Идентификатор должности	Число
rspostidname	Название должности	Строка (255)
personemail	E-mail (основной)	Строка (255)
pstatus	Статус пользователя (0 – Активен, 1 – Архив, 2 – Гость, 4 - Кандидат)	Малое
pextcode	Код внешней системы	Строка (255)

Информация о сессии входа в систему представляется следующей моделью данных:

Название поля	Описание	Тип
shost	Адрес входа	Строка
sip	IP-адрес, с которого произведен вход	Строка
slogintime	Дата входа	Дата
slogouttime	Дата завершения сессии	Дата

Добавление физического лица

Тип запроса: POST

/service/v2/persons

Входные параметры:

- Набор из параметров модели физического лица. Обязательный - pfirstname.

Выходные данные:

- Параметры из модели созданного физического лица

На заметку

- *Чтобы задать организацию, используйте либо параметр caid, либо caidname. При использовании параметра caidname будет выбрана существующая организация с указанным названием, если организации с таким названием нет, то она будет создана.*
- *Чтобы задать должность, используйте либо параметр rspostid, либо rspostidname. При использовании параметра rspostidname будет выбрана существующая должность с указанным названием, если должности с таким названием нет, то она будет создана.*

На практике

- *Если Вы регистрируете участников на мероприятия-вебинары, используйте метод Добавление участника мероприятия по E-mail.*
- *Если у всех физических лиц должен быть уникальный email, дополнительно передавайте email в поле **Логин**. Это позволит избежать случаев, когда может быть создано несколько пользователей с одинаковыми email-адресами.*

Получение информации о физическом лице

Тип запроса: GET

/service/v2/persons/{personId}

Параметр URL (обязательный):

- personId – идентификатор физического лица, по которому необходима информация.

Выходные данные:

- Параметры из модели созданного физического лица.

Получение информации о физическом лице по логину

Тип запроса: GET

/service/v2/persons/byLogin/{login}

Параметр URL (обязательный):

- login – логин физического лица, по которому необходима информация.

Выходные данные:

- Параметры из модели пользователя.

Изменение информации о физическом лице

Тип запроса: PUT

/service/v2/persons/{personId}

Параметр URL (обязательный):

- personId – идентификатор изменяемого пользователя.

Входные параметры:

- Набор из параметров модели физического лица, которые требуется изменить.

Удаление физического лица

Тип запроса: DELETE

/service/v2/persons/{personId}

Параметр URL (обязательный):

- personId – идентификатор удаляемого физического лица.

Получение списка сессий входа в систему

Тип запроса: GET

/service/v2/persons/{personId}/sessions

Параметр URL (обязательный):

- personId – идентификатор физического лица, по которому запрашивается информация.

Выходные данные:

- Массив объектов, состоящих из параметров модели сессии. Данные возвращаются постранично.

Получение последней сессии входа в систему

Тип запроса: GET

/service/v2/persons/{personId}/sessions/last

Параметр URL (обязательный):

- personId – идентификатор физического лица, по которому запрашивается информация.

Выходные данные:

- Набор из параметров модели сессии.

Модуль personGroups (Группы физических лиц)

Информация о группах физических лиц представляется следующей моделью данных:

Название поля	Описание	Тип
gcid	Идентификатор группы	Число
gcname	Название группы	Строка(100)
gcdesc	Описание группы	Строка(255)
gcparentid	Идентификатор родительской группы	Число
gcparentidname	Название родительской группы	Строка(100)
grkind	Вид группы (0 – динамическая, 1 – полудинамическая, 2 – статическая);	Малое

Создание группы физических лиц

Тип запроса: POST

/service/v2/personGroups

Входные параметры:

- Параметры из модели группы. Обязательное – gcname.

Выходные данные:

- Параметры из модели созданной группы.

На заметку

- *Если не указан вид группы, то по умолчанию создается динамическая группа.*
- *Всегда создается общая группа.*

Получение информации о группе физических лиц

Тип запроса: GET

/service/v2/personGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Параметры из модели группы.

Удаление группы физических лиц

Тип запроса: DELETE

/service/v2/personGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы

На заметку

– *Для удаления, группа не должна содержать дочерних групп.*

Изменение группы физических лиц

Тип запроса: PUT

/service/v2/personGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Входные параметры:

- Набор из параметров модели группы физических лиц, которые нужно изменить.

Выходные данные:

- Параметры из модели группы.

Добавление дочерней группы

Тип запроса: POST

/service/v2/personGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор добавляемой дочерней группы.

Исключение дочерней группы

Тип запроса: DELETE

/service/v2/personGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемой группы.

Получение списка дочерних групп

Тип запроса: GET

/service/v2/personGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели группы. Данные возвращаются постранично.

Добавление физического лица в группу

Тип запроса: POST

/service/v2/personGroups/{id}/childPersons/{personId}

Параметры URL (обязательные):

- id – идентификатор группы.
- personId – идентификатор физического лица.

На заметку

- *Добавлять элементы возможно только в группы вида «2» - статические.*
- *В полудинамические группы элементы добавляются фоновым заданием на основе настроек фильтра в группе, которые задаются через интерфейс системы.*
- *В динамические группы элементы не добавляются. Список элементов для отображения рассчитывается при обращении к группе на основе заданного фильтра.*
- *Настройка полудинамических и динамических групп через API не поддерживается.*

Исключение физического лица из группы

Тип запроса: DELETE

/service/v2/personGroups/{id}/childPersons/{personId}

Параметры URL (обязательные):

- id – идентификатор группы.
- personId – идентификатор исключаемого из группы физического лица.

Получение списка физических лиц, входящих в группу

Тип запроса: GET

/service/v2/personGroups/{id}/childPersons

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели физического лица. Данные возвращаются постранично.

Модуль roles (Доступ)

Информация о роли представляется следующей моделью данных:

Название поля	Описание	Тип
roleid	Идентификатор роли	Число
profileid	Идентификатор профиля роли	Число
rolename	Название	Строка (100)

sysrolename	Системное название	Строка (255)
roleisdefault	Присваивается данная роль по умолчанию пользователю при создании (true- да или false - нет)	Логическое

Добавление роли

Тип запроса: POST

/service/v2/roles

Входные параметры:

- Набор из параметров модели роли. Обязательными параметрами являются profileid, rolename, sysrolename.

Выходные данные:

- Параметры из модели созданной роли.

Получение информации о роли

Тип запроса: GET

/service/v2/roles/{id}

Параметр URL (обязательный):

- id – идентификатор роли.

Выходные данные:

- Параметры из модели роли.

Получение информации о роли по системному названию

Тип запроса: GET

/service/v2/roles/bySysName/{sysName}

Параметр URL (обязательный):

- sysName – системное название роли.

Выходные данные:

- Параметры из модели роли.

Получение информации о ролях пользователя

Тип запроса: GET

/service/v2/roles/byPerson/{personId}

Параметр URL (обязательный):

- personId – идентификатор пользователя.

Выходные данные:

- Массив из объектов модели роли.

Изменение информации о роли

Тип запроса: PUT

/service/v2/roles/{id}

Параметр URL (обязательный):
- id – id изменяемой роли.

Выходные данные:
- Параметры из модели роли.

Удаление роли

Тип запроса: DELETE

/service/v2/roles/{id}

Параметр URL (обязательный):
- id – идентификатор удаляемой роли.

Добавление роли пользователю

Тип запроса: POST

/service/v2/roles/{id}/addToPerson/{personId}

Параметры URL (обязательные):
- id – идентификатор роли;
- personId – идентификатор пользователя.

Удаление роли пользователя

Тип запроса: POST

/service/v2/roles/{id}/removeFromPerson/{personId}

Параметры URL (обязательные):
- id – идентификатор роли;
- personId – идентификатор физического лица.

Получение информации об организациях, за которые отвечает пользователь

Тип запроса: GET

service/v2/roles/byPerson/{personId}/{roleId}/ca

Метод позволяет получить список организаций, за которые ответственен пользователь с ролью на основе профиля **Внешний администратор** или **Администратор планирующей организации**.

Параметры URL (обязательные):
- personId – идентификатор физического лица.
- roleId – идентификатор роли.

Выходные данные:
- Массив объектов с параметрами организации. Данные возвращаются постранично.

Добавление организации, за которую отвечает пользователь

Тип запроса: POST

service/v2/roles/byPerson/{personId}/{roleId}/ca/{caId}

Метод позволяет добавить организацию в список организаций, за которые ответственен пользователь с ролью на основе профиля **Внешний администратор** или **Администратор планирующей организации**.

Параметры URL (обязательные):
- personId – идентификатор физического лица.
- roleId – идентификатор роли.
- caId – идентификатор организации.

Исключение организации, за которую отвечает пользователь

Тип запроса: DELETE

service/v2/roles/byPerson/{personId}/{roleId}/ca/{caId}

Метод позволяет исключить организацию из списка организаций, за которые ответственен пользователь с ролью на основе профиля **Внешний администратор** или **Администратор планирующей организации**.

Параметры URL (обязательные):
- personId – идентификатор физического лица.
- roleId – идентификатор роли.
- caId – идентификатор организации.

Назначение организации по умолчанию

Тип запроса: PUT

service/v2/roles/byPerson/{personId}/{roleId}/ca/{caId}/setDefault

Метод позволяет назначить организацию из списка организаций, за которые ответственен пользователь, как организацию по умолчанию для роли на основе профиля **Администратор планирующей организации**.

Параметры URL (обязательные):
- personId – идентификатор физического лица.
- roleId – идентификатор роли.
- caId – идентификатор организации.

Модуль cas (Организации)

Информация об организации представляется следующей моделью данных:

Название поля	Описание	Тип
caid	Идентификатор организации	Число
caname	Название	Строка (255)
caparentid	Идентификатор родительской организации	Число
cashortname	Короткое название организации	Строка (100)
castringcode	Код организации	Строка (255)

Добавление организации

Тип запроса: POST

/service/v2/cas

Входные параметры:

- Набор из параметров модели организации. Обязательным параметром является caname.

Выходные данные:

- Параметры из модели созданной организации

Получение информации об организации

Тип запроса: GET

/service/v2/cas/{id}

Параметр URL (обязательный):

- id – идентификатор организации.

Выходные данные:

- Параметры из модели организации.

Изменение информации об организации

Тип запроса: PUT

/service/v2/cas/{id}

Параметр URL (обязательный):

- id – идентификатор организации.

Входные параметры:

- Набор из параметров модели организации, которые требуется изменить.

Выходные данные:

- Параметры из модели организации.

Удаление организации

Тип запроса: DELETE

/service/v2/cas/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой организации.

Модуль caGroups (Группы организаций)

Информация о группах организаций представляется следующей моделью данных:

Название поля	Описание	Тип
gcid	Идентификатор группы	Число
gcname	Название группы	Строка(100)
gcdesc	Описание группы	Строка(255)
gcparentid	Идентификатор родительской группы	Число
gcparentidname	Название родительской группы	Строка(100)
grkind	Вид группы (0 – Динамическая, 1 – Полудинамическая, 2 – Статическая);	Малое

Создание группы организаций

Тип запроса: POST

/service/v2/caGroups

Входные параметры:

- Параметры из модели группы (обязательное – gcname).

Выходные данные:

- Параметры из модели созданной группы.

На заметку

- *Если не указан вид группы, то по умолчанию создается динамическая группа.*
- *Всегда создается общая группа.*

Получение информации о группе организаций

Тип запроса: GET

/service/v2/caGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Параметры из модели группы.

Удаление группы организаций

Тип запроса: DELETE

/service/v2/caGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

На заметку

– *Для удаления, группа не должна содержать дочерних групп.*

Изменение группы организаций

Тип запроса: PUT

/service/v2/caGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Входные параметры:

- Набор из параметров модели группы организаций, которые нужно изменить.

Выходные данные:

- Параметры из модели группы организаций.

Добавление дочерней группы

Тип запроса: POST

/service/v2/caGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы
- childId – идентификатор добавляемой группы

Исключение дочерней группы

Тип запроса: DELETE

/service/v2/caGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемой группы.

Получение списка дочерних групп

Тип запроса: GET

/service/v2/caGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели группы. Данные возвращаются постранично.

Добавление организации в группу

Тип запроса: POST

/service/v2/caGroups/{id}/childCAs/{caId}

Параметры URL (обязательные):
- id – идентификатор группы.
- caId – идентификатор организации.

На заметку

- *Добавлять элементы возможно только в группы вида «2» - статические.*
- *В полудинамические группы элементы добавляются фоновым заданием на основе настроек фильтра в группе, которые задаются через интерфейс системы.*
- *В динамические группы элементы не добавляются. Список элементов для отображения рассчитывается при обращении к группе на основе заданного фильтра.*
- *Настройка полудинамических и динамических групп через API не поддерживается.*

Исключение организации из группы

Тип запроса: DELETE

/service/v2/caGroups/{id}/childCAs/{caId}

Параметры URL (обязательные):
- id – идентификатор группы.
- caId – идентификатор организации.

Получение списка организаций, входящих в группу

Тип запроса: GET

/service/v2/caGroups/{id}/childCAs

Параметр URL (обязательный):
- id – идентификатор группы.

Выходные данные:
- Массив объектов с параметрами модели организации. Данные возвращаются постранично.

Модуль measures (Мероприятия)

Информация о мероприятии представляется следующей моделью данных:

Название поля	Описание	Тип
meid	Идентификатор мероприятия	Число
mename	Название	Строка (255)
medescription	Описание мероприятия	Число

metype	Тип мероприятия (0 – Типовое мероприятие, 1 – Мероприятие, 2 – Внутреннее мероприятие)	Строка (100)
mecode	Код мероприятия	Строка (255)
mestatus	Статус мероприятия (1 – В разработке, 2 – Активно, 3 – Отменено, 4 – В Архиве)	Малое
mestartdate	Дата и время начала мероприятия	Дата
meenddate	Дата и время окончания мероприятия	Дата
meeduform	Форма проведения (0 – Очная, 1 – Дистанционная, 2 – Смешанная)	Малое
mecontenttype	Вид мероприятия (0 – Электронный курс, 1 – Программа мероприятий, 3 – Практическое задание, 4 – Электронный тест, 5 – Вебинар, 6 – Очное мероприятие, 7 – Заочное мероприятие)	Малое
testid	Идентификатор электронного теста (только для мероприятий вида Электронный тест).	Число
testidname	Название электронного теста (только для мероприятий вида Электронный тест).	Строка (255)
ugrid	Идентификатор пользовательской группы ресурсов мероприятия	Число
ugridname	Название пользовательской группы ресурсов мероприятия	Строка (255)
mepasses	(Поле возвращается начиная в версии системы 4.6.50). Количество прохождений мероприятия. Для мероприятий всех типов кроме Вебинара – это количество участников мероприятия, которые имею статус прохождения "Пройдено". Для мероприятий типа Вебинар – это количество участников мероприятия, в результатах которых значение поля "Затраченное время" больше 0. Для типовых мероприятий берется сумма значений количества прохождений по всем дочерним мероприятиям.	Число

Информация об участнике мероприятия представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
mmid	Идентификатор участника мероприятия	Число
meid	Идентификатор мероприятия	Число
meidname	Название мероприятия	Строка(255)
personid	Идентификатор физического лица – участника мероприятия	Число
personidname	ФИО пользователя	Строка(302)
isaccess	Доступ (false – Закрыт, true- Открыт)	Логическое
mmfinishstatus	Статус завершения (0 – Не завершено, 1 – Завершено)	Малое

Информация о преподавателе мероприятия представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
mtid	Идентификатор преподавателя	Число
meid	Идентификатор мероприятия	Число
meidname	Название мероприятия	Строка(255)
personid	Идентификатор пользователя	Число
personidname	ФИО пользователя	Строка(302)
isaccess	Доступ (false – Закрыт, true- Открыт)	Логическое

Информация о записи вебинара представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
name	Название записи	Строка
period	Период ведения записи в формате [Дата и время начала записи] - [Дата и время завершения записи]	Строка
duration	Длительность записи в формате hh:mm:ss	Строка
viewLink	Ссылка для просмотра записи	Строка
downloadLink	Ссылка на скачивание записи в формате mvror	Строка
downloadVideoLink	Ссылка на скачивание mp4	Строка

Добавление мероприятия

Тип запроса: POST

/service/v2/measures

Входные параметры:

- Набор из параметров модели мероприятия. Обязательными параметрами являются mename, metype, meeduform.

На заметку

- *Для создания мероприятий-вебинаров (mecontenttype=5) в параметре metype указывайте «1», а в meeduform – «0».*
- *При создании, мероприятие по умолчанию создается со статусом Активно (статус мероприятия задается в параметре mestatus).*
- *При создании мероприятия через API в ведущие может добавляться создатель мероприятия. Эта логика настраивается в глобальных настройках системы на в блоке полей **Настройки мероприятий по умолчанию** – поле **При создании мероприятия автоматически делать создателя ведущим**.*

Выходные данные:

- Параметры из модели созданного мероприятия.

Получение информации о мероприятии

Тип запроса: GET

/service/v2/measures/{id}

Параметр URL (обязательный):

- id – идентификатор мероприятия.

Выходные данные:

- Параметры из модели мероприятия.

Изменение информации о мероприятии

Тип запроса: PUT

/service/v2/measures/{id}

Параметр URL (обязательный):

- id – идентификатор мероприятия.

Входные параметры:

- Набор из параметров модели мероприятия, которые требуется изменить.

Выходные данные:

- Параметры из модели мероприятия

Удаление мероприятия

Тип запроса: DELETE

/service/v2/measures/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого мероприятия.

Получение информации об участниках мероприятия

Тип запроса: GET

/service/v2/measures/{measureId}/members

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Массив из объектов модели участника мероприятий. Данные возвращаются постранично.

Добавление участника мероприятия

Тип запроса: POST

/service/v2/measures/{measureId}/members/{personId}

Параметры URL (обязательные):

- measureId – идентификатор мероприятия.
- personId – идентификатор физического лица.

Входные параметры:

- Возможно использовать параметр regtype – тип регистрации (0 – Саморегистрация, 1 – Назначение, 2 – Автоназначение, 3 – Назначение коллегой). По умолчанию используется значение **Саморегистрация**.

Выходные данные:

- Параметры из модели участника мероприятия.

Добавление участника мероприятия по E-mail

Тип запроса: POST

/service/v2/measures/{measureId}/members/regbyemail/{email}

Параметры URL (обязательные):

- measureId – идентификатор мероприятия;
- email – email регистрируемого на мероприятие пользователя.

Выходные данные:

- Параметры из модели пользователя.

На заметку

– *Данный метод аналогичен действию **Зарегистрировать** в списке участников мероприятия.*

– *При регистрации данным методом, указанный производится поиск физического лица с указанным E-mail по полю **Логин**. Если такое физическое лицо есть, производится регистрация этого физического лица на мероприятие. Если таких физических лиц нет, то создается новое и регистрируется на мероприятие.*

Добавление преподавателя мероприятия по E-mail

Метод поддерживается начиная с версии системы 4.6.50 2021.12.10.
Тип запроса: POST

/service/v2/measures/{measureId}/tutors/regbyemail/{email}

Параметры URL (обязательные):
- measureId – идентификатор мероприятия;
- email – email регистрируемого на мероприятие пользователя.

Входные параметры (необязательные):
- Параметр sendNotifications для отправки уведомления о регистрации на мероприятие.Значение по умолчанию: «true».
- Параметр addRolesByDefault для добавления ролей по умолчанию при создании физического лица в случае отсутствия физического лица в системе. Значение по умолчанию: «true».
- Параметр enableSearchByEmail для дополнительного поиска физического лица в системе по адресу электронной почты (personemail) вместо логина (pilogin) по умолчанию. Значение по умолчанию: «false».

Выходные данные:
- id физического лица, найденного либо созданного и зарегистрированного на мероприятие, как преподаватель.

На заметку

– *Данный метод с параметрами по умолчанию аналогичен действию **Зарегистрировать** в списке преподавателей мероприятия.*

Исключение участника мероприятия

Тип запроса: DELETE

/service/v2/measures/{measureId}/members/{personId}

Параметры URL (обязательные):
- measureId – идентификатор мероприятия.
- personId – идентификатор физического лица.

Отправка приглашения пользователю на мероприятие

Тип запроса: PUT

/service/v2/measures/{measureId}/members/invite/{personId}

На заметку

Данный метод аналогичен действию «Пригласить» в списке участников мероприятия-вебинара.

Параметры URL (обязательные):

- measureId – идентификатор мероприятия.
- personId – идентификатор физического лица.

Получение информации о преподавателях мероприятия

Тип запроса: GET

/service/v2/measures/{measureId}/tutors

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Массив из объектов модели преподавателя мероприятия. Данные возвращаются постранично.

Добавление преподавателя на мероприятие

Тип запроса: POST

/service/v2/measures/{measureId}/tutors/{personId}

Параметры URL (обязательные):

- measureId – идентификатор мероприятия.
- personId – идентификатор физического лица.

Выходные данные:

- Параметры из модели преподавателя.

Удаление преподавателя мероприятия

Тип запроса: DELETE

/service/v2/measures/{measureId}/tutors/{personId}

Параметры URL (обязательные):

- measureId – идентификатор мероприятия.
- personId – идентификатор физического лица.

Получение ссылки гостевого входа в вебинар

Тип запроса: GET

/service/v2/measures/{measureId}/webinarAnonymousLink

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Ссылка для анонимного входа на вебинар.

Получение ссылки для входа участника мероприятия в вебинар

Тип запроса: GET

/service/v2/measures/{measureId}/webinarLink/{personId}

Параметры URL (обязательные):

- measureId – идентификатор мероприятия.
- personId – идентификатор участника.

Выходные данные:

- Ссылка для входа на вебинар участника.

Получение ссылок на записи вебинара

Тип запроса: GET

/service/v2/measures/{measureId}/webinarRecords

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Параметры модели записи вебинара.

Получение результатов участия в мероприятии

Тип запроса: GET

/service/v2/measures/{measureId}/results

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Массив объектов результатов участия в мероприятии. Данные возвращаются постранично.

На заметку

– *Функция возвращает только результаты зачислений, у которых были попытки прохождения.*

Изменение дат обучения слушателя на мероприятии

Тип запроса: PUT

/service/v2/measures/members/{mmId}/changeDates

Параметр URL (обязательный):

- mmId – идентификатор участника мероприятия.

Входные параметры:
- startDate – дата начала обучения слушателя на мероприятии.
- endDate – дата окончания обучения слушателя на мероприятии.

Изменение дат обучения слушателя на мероприятии, зарегистрированного по источнику

Тип запроса: PUT

/service/v2/measures/members/{mmId}/{sourceId}/changeDates

Параметры URL (обязательные):
- mmId – идентификатор участника мероприятия. Используется идентификатор из мероприятия, даты обучения по которому нужно изменить.
- sourceId – идентификатор программы мероприятий, которая является источником зачисления участника на мероприятие.

Входные параметры:
- startDate – дата начала обучения слушателя на мероприятии.
- endDate – дата окончания обучения слушателя на мероприятии.

Получение компонентов программы мероприятия

Тип запроса: GET

/service/v2/measures/{measureId}/components

Параметр URL (обязательный):
- measureId – идентификатор мероприятия (программы мероприятий).

Выходные данные:
- Массив объектов с параметрами модели мероприятия.

Получение списка ресурсов мероприятия

Тип запроса: GET

/service/v2/measures/{measureId}/resources

Параметр URL (обязательный):
- measureId – идентификатор мероприятия.

Выходные данные:
- Массив объектов с параметрами ресурса.

Завершение мероприятия

Тип запроса: POST

/service/v2/measures/{measureId}/close

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Получение списка вебинар-опросов мероприятия

Тип запроса: GET

/service/v2/measures/{measureId}/surveys

Параметр URL (обязательный):

- measureId – идентификатор мероприятия.

Выходные данные:

- Массив объектов с параметрами вебинар-опроса.

Модуль myMeasures (Мои мероприятия)

Информация о мероприятиях пользователя представляется следующей моделью данных:

Название поля	Описание	Тип
meid	Идентификатор мероприятия	Число
meidname	Название мероприятия	Строка
mecontenttype	Вид мероприятия (0 – Электронный курс, 1 – Программа мероприятий, 3 – Практическое задание, 4 – Электронный тест, 5 – Вебинар, 6 – Очное мероприятие, 7 – Заочное мероприятие)	Малое
sourceid	Идентификатор программы мероприятий, по которой произведено зачисление на текущее мероприятие (источник).	Число
mmid	Идентификатор участника	Число
mrstatusid	Статус прохождения	Строка
mmfinishstatus	Статус завершения (0 – Не завершено, 1 – Завершено)	Малое
isaccess	Доступ (false – Закрыт, true- Открыт)	Логическое
mmdregistrationtype	Тип регистрации участника на мероприятие (0 – Саморегистрация, 1 – Назначение, 2 – Автоназначение, 3 – Назначение коллегой)	Малое
mrscoreinpercent	Оценка в процентах	Число
result	Набранный балл	Число
mmdstartdate	Дата начала обучения по текущему источнику	Дата
mmdenddate	Дата окончания обучения по текущему источнику	Дата
contentid	Идентификатор электронного курса (значение возвращается только для мероприятий вида Электронный курс)	Число
testcsid	Идентификатор раздела контента, с котором связываются результаты по прохождению теста (значение возвращается только для мероприятий вида Электронный тест)	Число
testid	Идентификатор электронного теста (значение возвращается только для мероприятий вида Электронный тест)	Число
link	Параметр используется для обратной совместимости.	Строка

Информация о мероприятиях-вебинарах участника представляется следующей моделью данных:

Название поля	Описание	Тип
meid	Идентификатор мероприятия	Число
meidname	Название мероприятия	Строка
mestartdate	Начало мероприятия	Дата
meenddate	Окончание мероприятия	Дата
link	Ссылка для входа в вебинар	Строка

Получение списка мероприятий-вебинаров участника

Тип запроса: GET

/service/v2/myMeasures/{personId}/webinars

Метод возвращает список мероприятий-вебинаров участника со ссылкой для входа в вебинар.

Параметр URL (обязательный):

- personId – идентификатор физического лица.

Выходные данные:

- Массив объектов модели мероприятия-вебинара участника.

Получение списка мероприятий пользователя в качестве участника

Тип запроса: GET

/service/v2/myMeasures/{personId}/member

Параметр URL (обязательный):

- personId – идентификатор физического лица.

Выходные данные:

- Массив объектов модели участника мероприятия.

Получение списка мероприятий пользователя в качестве преподавателя

Тип запроса: GET

/service/v2/myMeasures/{personId}/tutor

Параметр URL (обязательный):

- personId – идентификатор физического лица.

Выходные данные:

- Массив объектов модели преподавателя мероприятия.

Получение списка мероприятий авторизованного пользователя

Тип запроса: GET

/service/v2/myMeasures/{personId}/measures

Параметр URL (обязательный):

- personId – идентификатор пользователя.

Входные параметры:

- usrsesid– id сессии пользователя, генерируемый методом Открытие пользовательской сессии. **Обязательный параметр**.

Выходные данные:

- Массив объектов модели мероприятий пользователя.
- Link – ссылка для запуска электронного теста или курса.

Модуль measureResults (Результаты участия в мероприятии)

Информация о результате участия в мероприятии представляется следующей моделью данных:

Название поля	Описание	Тип
mmid	Идентификатор участника мероприятия	Число
mrstatusid	Статус прохождения (0 – Не сдано, 1 – Пройдено, 2 – Просмотрен (не используется), 3 – В процессе, 4 – Не начат, 5 – На проверке, 6 – проверен)	Малое
mrlastaccesstime	Время начала последней попытки	Дата
mrscoreinpercent	Набранная оценка в процентах	Число
mrscore	Набранный балл	Число
mrprocent	Процент прохождения	Число дробное (от 0 до 1)
mrduration	Общая продолжительность всех попыток в секундах	Число дробное
mrstarttime	Время начала первой попытки	Дата
mrendtime	Время окончания последней попытки	Дата
mrattemptcount	Общее количество попыток	Число

Информация о результатах участия в мероприятии по разделам электронного контента представляется следующей моделью данных:

Название поля	Описание	Тип
mmid	Идентификатор участника мероприятия	Число
csid	Идентификатор раздела электронного контента мероприятия	Число
csname	Название раздела электронного контента мероприятия	Строка
meid	Идентификатор мероприятия	Число
rcsresourceid	Идентификатор ресурса медиатеки (возвращается только для разделов контента, в которых добавлен ресурс-файл медиатеки)	Число
cssectiontype	Вид раздела (0 – SCO-Пакет, 1- Тест, 2 - Папка, 3 - Файл, 4 - Ссылка, 5 - HTML, 6 - Ресурс, 7 - HTML-контейнер, 8 - Опрос)	Малое
isrequeried	Показывает, является ли раздел обязательным для прохождения	Логическое
prstatusid	Статус прогресса (0 – Не сдано, 1 – Пройдено, 2 – Просмотрен (не используется), 3 – В процессе, 4 – Не начат, 5 – На проверке, 6 – Проверен)	Число
attemptlimit	Максимальное количество попыток	Число
prscore	Набранный балл	Число с дробью
passscore	Проходной балл	Число с дробью
filelink	Ссылка для скачивания оригинального файла медиатеки (возвращается только для разделов контента, в которых добавлен ресурс-файл медиатеки). Файл скачивается с учетом доступа пользователя на скачивание. Для скачивания необходимо авторизовать пользователя и добавить к ссылке параметр usrsesid со значением, полученным методом Открытие пользовательской сессии. Скачивание файла по этой ссылке не засчитывает прохождение раздела курса.	Строка
previewfilelink	Ссылка для скачивания конвертированной версии файла медиатеки для предпросмотра в браузере (возвращается только для разделов контента, в которых добавлен ресурс-файл медиатеки и если у файла есть конвертированная версия). Файл скачивается с учетом доступа пользователя на скачивание. Для скачивания необходимо авторизовать пользователя и добавить к ссылке параметр usrsesid со значением, полученным методом Открытие пользовательской сессии. Скачивание файла по этой ссылке не засчитывает прохождение раздела курса.	Строка
link	Ссылка для запуска раздела электронного контента на сервере доставки. Ссылка возвращается для разделов всех типов, но работает только для разделов вида тест или SCO – Пакет.	Строка

Добавление результата

Тип запроса: POST

/service/v2/measureResults

Входные параметры:

- Параметры из модели результата участия в мероприятии (обязательное поле - mmid).

Выходные данные:

- Параметры из модели результата участия в мероприятии.

Получение информации о результате

Тип запроса: GET

/service/v2/measureResults/{mmid}

Параметр URL (обязательный):

- mmid – идентификатор участника мероприятия.

Выходные данные:

- Параметры из модели результата участия в мероприятии.

На заметку

- *Результат возвращается, только если были попытки прохождения мероприятия участником. До попытки прохождения мероприятия результат не формируется.*
- *Если результат не был сформирован, при запросе по API возвращается сообщение* ***Not found object by id:{mmid}***

Изменение результата

Тип запроса: PUT

/service/v2/measureResults/{mmid}

Параметр URL (обязательный):

- mmid – идентификатор участника мероприятия.

Входные параметры:

- Параметры из модели результата участия в мероприятии.

Выходные данные:

- Параметры из модели результата участия в мероприятии.

Удаление результата

Тип запроса: DELETE

/service/v2/measureResults/{mmid}

Параметр URL (обязательный):

- mmid – идентификатор участника мероприятия.

Получение информации о результатах по разделам

Тип запроса: GET

/service/v2/measureResults/{mmid}/progresses

Параметр URL (обязательный):

- mmid – идентификатор участника мероприятия.

Выходные данные:

- Параметры из модели результата участия в мероприятии по разделам электронного контента.

На заметку

– *Если требуется сохранить данные о прохождении разделов курса при скачивании ресурсов по ссылкам **previewfilelink** и **filelink**, нужно создать попытку методом Добавление попытки, передав значения **mmid** и **csid**, полученные для раздела с ресурсом, а также другие данные попытки.*

Модуль measureProgress (Прогресс участия в мероприятии)

Информация о прогрессе участия (результат по разделу электронного контента участника мероприятия) в мероприятии представляется следующей моделью данных:

Название поля	Описание	Тип
prid	Идентификатор прогресса	Число
mmid	Идентификатор участника мероприятия	Число
prprocent	Процент прохождения	Число от 0 до 1
prscore	Набранный балл	Число
prstatusid	Статус прохождения (0 – Не сдано, 1 – Сдано, 2 – Просмотрен (не используется), 3 – В процессе, 4 – Не начат, 5 – На проверке, 6 – проверен)	Малое
prstarttime	Время начала первой попытки	Дата
prendtime	Время окончания последней попытки	Дата
prattemptcount	Общее количество попыток	Число
prduration	Общая продолжительность всех попыток в секундах	Число дробное
csid	Идентификатор раздела электронного контента, с которым связан прогресс	Число

Добавление прогресса

Тип запроса: POST

/service/v2/measureProgress

Входные параметры:

- Параметры из модели прогресса.

Выходные данные:

- Параметры созданного прогресса.

На заметку

– *Прогресс создается автоматически при создании попытки методом Добавление попытки, передав значения mmid (идентификатор участника мероприятия) и csid (идентификатор раздела электронного контента).*

Получение информации о прогрессе

Тип запроса: GET

/service/v2/measureProgress/{prid}

Параметр URL (обязательный):

- prid – идентификатор прогресса.

Выходные данные:

- Параметры из модели прогресса.

Изменение прогресса

Тип запроса: PUT

/service/v2/measureProgress/{prid}

Параметр URL (обязательный):

- prid – идентификатор прогресса.

Входные параметры:

- Параметры из модели прогресса.

Выходные данные:

- Параметры из модели прогресса.

Удаление прогресса

Тип запроса: DELETE

/service/v2/measureProgress/{prid}

Параметр URL (обязательный):

- prid – идентификатор прогресса.

Модуль measureAttempts (Попытки прохождения мероприятия)

Информация о попытке участия в мероприятии представляется следующей моделью данных:

Название поля	Описание	Тип
attid	Идентификатор попытки	Число
prid	Идентификатор прогресса	Число
attprocent	Процент прохождения	Число от 0 до 1
attscore	Набранный балл	Число

attstatusid	Статус прохождения (0 – Не сдано, 1 – Пройдено, 2 – Просмотрен (не используется), 3 – В процессе, 4 – Не начат, 5 – На проверке, 6 – Проверен)	Малое
attstarttime	Время начала попытки	Дата
attendtime	Время окончания попытки	Дата
attattemptcount	Номер попытки (при создании попытки заполняется автоматически)	Число
attduration	Продолжительность прохождения попытки в секундах	Число дробное
attscoreinpercent	Оценка в процентах по попытке	Число дробное

Добавление попытки

Тип запроса: POST

/service/v2/measureAttempts

Входные параметры:

- Параметры из модели попытки.
- При создании попытки возможно указать параметр csid (идентификатор раздела электронного контента) вместе с mmid, при этом автоматически будет создан прогресс для участника по разделу контента, передаваемому в csid.

Выходные данные:

- Параметры из модели попытки.

Получение информации о попытке

Тип запроса: GET

/service/v2/measureAttempts/{attid}

Параметр URL (обязательный):

- attid – идентификатор попытки.

Выходные данные:

- Параметры из модели попытки.

Изменение попытки

Тип запроса: PUT

/service/v2/measureAttempts/{attid}

Параметр URL (обязательный):

- attid – идентификатор попытки.

Входные параметры:

- Параметры из модели попытки.

Выходные данные:

- Параметры из модели попытки.

Удаление попытки

Тип запроса: DELETE

/service/v2/measureAttempts/{attid}

Параметр URL (обязательный):

- attid – идентификатор попытки.

Модуль measureInteractions (Данные прохождения тестирований)

Информация о сохраненном ответе на вопрос тестирования по попытке участия в мероприятии представляется следующей моделью данных:

Название поля	Описание	Тип
inid	Идентификатор ответа на вопрос по попытке	Число
attid	Идентификатор попытки.	Число
inresult	Набранный балл	Число
invalue	Идентификатор ответа на вопрос. Для вопросов разных типов ответ сохраняется в разном формате: • Одиночный выбор - <идентификатор ответа>. Передается только идентификатор выбранного ответа. • Множественный выбор - <идентификатор ответа>,<идентификатор ответа>. Передается список идентификаторов данных ответов через запятую. • Порядок - <идентификатор ответа>,<идентификатор ответа>. Указывается список идентификаторов ответа через запятую в порядке, указанном пользователем. • Соответствие - <идентификатор ответа>.<идентификатор выбранного пользователем соответствия>,<идентификатор ответа>.<идентификатор выбранного пользователем соответствия>. Указывается идентификатор ответа, через точку указывается идентификатор выбранного соответствия. Разные пары «ответ.выбранное соответствие» разделяются запятыми.	Число
qid	Идентификатор вопроса теста. Идентификатор возвращается методом Получение данных тестирования и имеет вид **A$245559**.	Строка
qidname	Название вопроса теста	Строка
inorder	Порядок показа вопроса пользователю	Число
indate	Дата ответа на вопрос	Дата
intime	Время, затраченное на ответ (сек)	Число
incomment	Комментарий к ответу	Дата
inmaxpoint	Максимально баллов за ответ на вопрос	Число

Добавление ответа участника на вопрос

Тип запроса: POST

/service/v2/measureInteractions

Входные параметры:

- Параметры из модели ответа участника на вопрос.

На заметку

– *При добавлении ответа на вопрос необходимо передавать идентификатор попытки (attid), набранный балл (inresult), идентификатор ответы на вопрос (invalue), идентификатор вопроса теста (qid), максимально баллов ответов за вопрос (inmaxpoint).*

Выходные данные:

- Параметры из модели ответа участника на вопрос.

Получение информации об ответе участника на вопрос

Тип запроса: GET

/service/v2/measureInteractions/{inid}

Параметр URL (обязательный):

- inid – идентификатор ответа участника на вопрос.

Выходные данные:

- Параметры из модели ответа участника на вопрос.

Изменение ответа участника на вопрос

Тип запроса: PUT

/service/v2/measureInteractions/{inid}

Параметр URL (обязательный):

- inid – идентификатор ответа участника на вопрос.

Входные параметры:

- Параметры из модели ответа участника на вопрос.

Выходные данные:

- Параметры из модели ответа участника на вопрос.

Удаление ответа участника на вопрос

Тип запроса: DELETE

/service/v2/measureInteractions/{inid}

Параметр URL (обязательный):

- iInid – идентификатор ответа участника на вопрос.

Модуль measureGroups (Группы мероприятий)

Информация о группах мероприятий представляется следующей моделью данных:

Название поля	Описание	Тип
gcid	Идентификатор группы	Число
gcname	Название группы	Строка(100)
gcdesc	Описание группы	Строка(255)

gcparentid	Идентификатор родительской группы	Число
gcparentidname	Название родительской группы	Строка(100)
grkind	Вид группы (0 – Динамическая, 1 – Полудинамическая, 2 – Статическая);	Малое

Создание группы мероприятий

Тип запроса: POST

/service/v2/measureGroups

Входные параметры:

- Параметры из модели группы. Обязательное – gcname.

Выходные данные:

- Параметры из модели созданной группы.

На заметку

- *Если не указан вид группы, то по умолчанию создается динамическая группа.*
- *Всегда создается общая группа.*

Получение информации о группе мероприятий

Тип запроса: GET

/service/v2/measureGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Параметры из модели группы.

Удаление группы мероприятий

Тип запроса: DELETE

/service/v2/measureGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

На заметку

- *Для удаления, группа не должна содержать дочерних групп.*

Изменение группы мероприятий

Тип запроса: PUT

/service/v2/measureGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Входные параметры:

- Набор из параметров модели группы мероприятий, которые нужно изменить.

Выходные данные:

- Параметры из модели группы мероприятий.

Добавление дочерней группы

Тип запроса: POST

/service/v2/measureGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор добавляемой группы.

Исключение дочерней группы

Тип запроса: DELETE

/service/v2/measureGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемой группы.

Получение списка дочерних групп

Тип запроса: GET

/service/v2/measureGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели группы. Данные возвращаются постранично.

Добавление мероприятия в группу

Тип запроса: POST

/service/v2/measureGroups/{id}/childMeasures/{measureId}

Параметры URL (обязательные):

- id – идентификатор группы.
- measureId – идентификатор мероприятия.

На заметку

- *Добавлять элементы возможно только в группы вида «2» - статические.*
- *В полудинамические группы элементы добавляются фоновым заданием на основе настроек фильтра в группе, которые задаются через интерфейс системы.*
- *В динамические группы элементы не добавляются. Список элементов для отображения рассчитывается при обращении к группе на основе заданного фильтра.*
- *Настройка полудинамических и динамических групп через API не поддерживается.*

Исключение мероприятия из группы

Тип запроса: DELETE

/service/v2/measureGroups/{id}/childMeasures/{measureId}

Параметры URL (обязательные):

- id – идентификатор группы.
- measureId – идентификатор мероприятия.

Получение списка мероприятий, входящих в группу

Тип запроса: GET

/service/v2/measureGroups/{id}/childMeasures

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели мероприятия. Данные возвращаются постранично.

Модуль measureUserGroups (Пользовательские группы мероприятий)

Информация о пользовательских группах мероприятий представляется следующей моделью данных:

Название поля	Описание	Тип
ugrid	Идентификатор пользовательской группы	Число
ugrname	Название пользовательской группы	Строка
ugrparentid	Идентификатор родительской пользовательской группы. Обязательное поле. Если нужно разместить группу «в корне» пользовательских групп, нужно указать значение «-10», соответствующее системной пользовательской группе «ПКМ».	Число
ugrparentidname	Название родительской пользовательской группы	Строка
ugrdesc	Стандартное описание пользовательской группы	Строка
ugrextdesc	Расширенное описание пользовательской группы	Строка
personid	Идентификатор создателя пользовательской группы	Число
personidname	ФИО создателя пользовательской группы	Строка
ugrcreatedate	Дата создания пользовательской группы	Дата

ugrshowdesc	Показывать описания группы (0 – Не показывать, 1 – Показывать стандартное, 2 – Показывать расширенное)	Малое
ugrshowsubgroup	Показывать подгруппы (0 – Не показывать, 1 – Первого уровня, 2 – Всех подуровней, 3 - заданное число, 4 – Показывать подгруппы как списки с категориями, 5 – Показать, как список, 6 – Показать подгруппы как списки без категорий)	Малое
ugrshowsubgroupimage	Показывать изображения подгрупп	Логическое
ugrshowsubgroupdesc	Показывать описания подгрупп (0 – Не показывать, 1 – Показывать стандартные, 2 – Показывать расширенные)	Малое
ugrsubgrouplevelcount	Количество показываемых подуровней	Число
ugrorder	Порядок	Число
ugrcolumn	Количество колонок	Число
ugrsortorder	Сортировать элементы (0 – По алфавиту, 1 – По дате добавления)	Малое
ugrsortdirection	Порядок сортировки (0 – По возрастанию, 1 – По убыванию)	Малое
objcount	Количество объектов в пользовательской группе (дочерние пользовательские группы не учитываются)	Число

Добавление пользовательской группы

Тип запроса: POST

/service/v2/mediaUserGroups

Входные параметры:

- Набор из параметров модели пользовательской группы. Обязательным параметром является название и родительская пользовательская группа.

Выходные данные:

- Параметры из модели созданной пользовательской группы.

Получение информации о пользовательской группе

Тип запроса: GET

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Параметры из модели пользовательской группы.

Изменение информации о пользовательской группе

Тип запроса: PUT

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Входные параметры:

- Набор из параметров модели пользовательской группы, которые требуется изменить.

Выходные данные:

- Параметры из модели пользовательской группы

Удаление пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой пользовательской группы.

Добавление дочерней пользовательской группы

Тип запроса: POST

/service/v2/mediaUserGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор добавляемой дочерней пользовательской группы.

Исключение дочерней пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор исключаемой дочерней пользовательской группы.

Получение списка дочерних пользовательских групп

Тип запроса: GET

/service/v2/mediaUserGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Массив объектов с параметрами из модели пользовательской группы.

Добавление мероприятия в пользовательскую группу

Тип запроса: POST

/service/v2/mediaUserGroups/{id}/childObjects/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор добавляемого мероприятия.

Исключение мероприятия из пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}/childObjects/{childId}

Параметры URL (обязательный):

- id – идентификатор пользовательской группы.
- childId – идентификатор исключаемого мероприятия.

Получение списка мероприятий пользовательской группы

Тип запроса: GET

/service/v2/mediaUserGroups/{id}/childObjects

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Массив объектов модели мероприятия.

Модуль surveys (Вебинар-опросы)

Модуль предназначен для получения результатов по опросам, проведенным в виртуальной комнате (вебинар-опросы). Для получения списка вебинар-опросов мероприятия нужно использовать метод Получение списка вебинар-опросов мероприятия.

Информация о вебинар-опросе представляется следующей моделью данных:

Название поля	Описание	Тип
mvid	Идентификатор опроса	Число
meid	Идентификатор мероприятия, в которое добавлен опрос	Число
personid	Идентификатор физического лица, создавшего опрос	Число
personidname	ФИО физического лица, создавшего опрос	Строка
mvname	Название опроса	Строка
mvtype	Тип опроса (0 - Одиночный выбор; 1- Множественный выбор)	Число
mvstatus	Статус проведения опроса (0 – Не начат; 1 - Проведен)	Малое
mvisshowresults	Показывать результаты участникам во время опроса	Логическое
mvisfreeanswer	Свой вариант (false – Не разрешен; true- Разрешен)	Логическое
mvopen	Открыт ли опрос в виртуальной комнате. Параметр отражает, открыт ли опрос в виртуальной комнате в окне Опрос (false – Не открыт; true - Открыт).	Логическое
mvstarted	Запущен ли опрос в виртуальной комнате. Параметр отражает, запущено ли проведение опроса в виртуальной комнате (false – Не запущен; true - Запущен).	Логическое

Информация о варианте ответа на вебинар-опрос представляется следующей моделью данных:

Название поля	Описание	Тип
mvarid	Идентификатор варианта ответа на опрос	Число

mvid	Идентификатор опроса	Число
mvidname	Название опроса	Строка
mvartext	Текст варианта ответа	Строка

Информация о статистике по ответам на вебинар-опрос представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
variant	Текст ответа на опрос	Текст
freevariant	Признак, является ли ответ свободным ответом на опрос	Логическое
count	Количество ответов пользователей, в которых был выбран вариант ответа. Для опросов типа Множественный выбор несколько вариантов ответа могут быть выбраны в одном ответе пользователя, поэтому общее количество ответов может быть больше чем количество пользователей, ответивших на опрос.	Число
procent	Процент ответов пользователей, в которых был выбран вариант ответа. Для опросов типа Множественный выбор несколько вариантов ответа могут быть выбраны в одном ответе пользователя, поэтому сумма процентов во всех вариантах ответа на опрос может быть больше 100%.	Текст

Информация об ответе пользователя на вебинар-опрос представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
maid	Идентификатор ответа пользователя на опрос	Число
answer	Текст ответа на опрос. В опросах типа множественный выбор через точку с запятой перечисляется текст данных вариантов ответа.	Строка
freeanswer	Текст свободного ответа на опрос. Пустое значение, если в ответе пользователя не было свободного ответа. В опросах типа множественный выбор может возвращаться вместе со значением в поле answer.	Строка
madate	Дата и время ответа на опрос	Дата
personid	Идентификатор физического лица, ответившего на опрос	Число
personidname	ФИО физического лица, ответившего на опрос	Строка

Информация о свободном ответе пользователя на вебинар-опрос представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
maid	Идентификатор ответа пользователя на опрос	Число
mvid	Идентификатор опроса	Число
mvidname	Название опроса	Строка
matext	Текст свободного ответа	Строка
mavaridlist	Идентификаторы ответов на опрос, данных вместе со свободным ответом. Возвращаются в виде строки чисел, разделенных запятой.	Строка
madate	Дата и время ответа на опрос	Дата
personid	Идентификатор физического лица, ответившего на опрос	Число
personidname	ФИО физического лица, ответившего на опрос	Строка

Добавление вебинар-опроса

Тип запроса: POST

/service/v2/surveys

Входные параметры:

- Набор из параметров модели вебинар-опроса кроме mvopen и mvstarted.

Выходные данные:

- Идентификатор созданного вебинар-опроса.

Получение информации о вебинар-опросе

Тип запроса: GET

/service/v2/surveys/{id}

Параметр URL (обязательный):

- id – идентификатор вебинар-опроса.

Выходные данные:

- Параметры из модели вебинар-опроса.

Изменение информации о вебинар-опросе

Тип запроса: PUT

/service/v2/surveys/{id}

Параметр URL (обязательный):

- id – идентификатор вебинар-опроса.

Входные параметры:

- Набор из параметров модели вебинар-опроса, которые требуется изменить, кроме mvopen и mvstarted.

Удаление вебинар-опроса

Тип запроса: DELETE

/service/v2/surveys/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого вебинар-опроса.

Получение списка ответов на вебинар-опрос

Тип запроса: GET

/service/v2/surveys/{surveyId}/answers

Параметр URL (обязательный):

- surveyId – идентификатор вебинар-опроса.

Выходные данные:

- Массив объектов с параметрами варианта ответа на вебинар-опрос.

Получение статистики по ответам на вебинар-опрос

Тип запроса: GET

/service/v2/surveys/{surveyId}/byanswer

Параметр URL (обязательный):

- surveyId – идентификатор вебинар-опроса.

Выходные данные:

- Массив объектов с параметрами статистики по ответам на вебинар-опрос.

Получение ответов пользователей на вебинар-опрос

Тип запроса: GET

/service/v2/surveys/{surveyId}/byperson

Параметр URL (обязательный):

- surveyId – идентификатор вебинар-опроса.

Выходные данные:

- Массив объектов с параметрами ответа пользователя на вебинар-опрос.

Получение свободных ответов пользователей на вебинар-опрос

Тип запроса: GET

/service/v2/surveys/{surveyId}/freeanswers

Параметр URL (обязательный):

- surveyId – идентификатор вебинар-опроса.

Выходные данные:

- Массив объектов с параметрами свободного ответа пользователя на вебинар-опрос.

Модуль mysurvey (Мои опросы)

Модуль предназначен для получения списка опросов, назначенных пользователю, авторизация которого передана в API, а также получения данных опроса для прохождения опроса на внешнем сайте и добавления прохождения опроса с внешнего сайта в систему Мираполис. Поддерживается как прохождение опросов, выводящихся в списке Мои опросы, так и прохождение опросов, которые являются разделами электронных курсов.

Информация об опросах, назначенных текущему пользователю, представляется следующей моделью данных:

Название поля	Описание	Тип
id	Идентификатор опроса	Строка
name	Название опроса	Строка
meid	Идентификатор мероприятия	Строка
mename	Название мероприятия	Строка
personid	Идентификатор пользователя, заполняющего опрос	Строка
personname	ФИО пользователя, заполняющего опрос	Строка
memberid	Идентификатор пользователя, на которого заполняется опрос	Строка
member	ФИО пользователя, на которого заполняется опрос	Строка
surtype	Тип опроса: • «0» - «Стандартный»	Строка

	• «1» - «По мероприятию» • «2» – «Как элемент курса» • «3» – «По преподавателям» • «4» – «Обратная связь по слушателям мероприятия» • «5» – «Проверка» • «6» – «Рекомендации для подбора» • «7» – «Опрос подбора»	
pastatus	Статус: • «0» - Не заполнен • «1» - Заполнен • «2» - Не отправлен	Строка

Данные опроса для заполнения пользователем представляются следующей моделью данных:

Название поля	**Описание**	**Тип**
surid	Идентификатор опроса	Строка
surname	Название опроса	Строка
surcode	Код	Строка
surdescription	Описание	Строка
surissection	Отображать опрос с разделами («true» – с разделами, «false» – без разделов)	Логическое
votesWithAnswer	Вопросы с вариантами ответов. Заполнено, если параметр «surissection» = «false». В этом случае все вопросы возвращаются в массиве **votesWithAnswer**, а массив **section** – пустой.	Массив
section	Массив разделов опроса. Заполнен, если параметр «surissection» = «true». В этом случае данные опроса возвращаются в массиве **section**, а массив **votesWithAnswer** опроса– пустой.	Массив

Структура данных в массиве **section**. Массив **section** заполнен, когда опрос настроен на отображение с разделами («surissection» = «true»), иначе массив **section** возвращается пустым. В массиве возвращаются данные по каждому разделу и массив вопросов раздела опроса:

Название поля	**Описание**	**Тип**
sursid	Идентификатор раздела	Строка
sursname	Название раздела	Строка
sursorder	Порядок раздела	Число
votesWithAnswer	Массив вопросов раздела опроса с вариантами ответа.	Массив

Структура данных в массиве **votesWithAnswer**. Структура данных массива одинаковая как при получении массива votesWithAnswer в объекте опроса так и при получении в массиве разделов section. По каждому вопросу опроса возвращается следующий набор данных:

Название поля	**Описание**	**Тип**
vote	Вопрос. Содержит описание вопроса. Описание полей в таблице **vote**.	Объект
variantAnswer	Варианты ответа на вопрос.	Массив
answer	Выбранный ответ на вопрос. Возвращается для вопросов, для которых были сохранены ответы.	Массив

Структура данных объекта вопрос **vote**:

Название поля	**Описание**	**Тип**
voteid	Идентификатор вопроса	Строка
votequestion	Текст вопроса	Строка
surid	Идентификатор опроса	Строка
suridname	Название опроса	Строка
votehasfree	Свободный ответ («true» – возможен свободный ответ, «false» – нет)	Логическое
votetype	Тип вопроса: • «0» – «Одиночный выбор» • «1» – «Множественный выбор»	Малое число

	• «2» – «Свободный ответ (текст)» • «3» – «Ранжирование» • «4» – «Свободный ответ (число)» • «5» – «Дата» • «6» – «Дата и время» • «7» – «Выбор из диапазона» • «8» – «Файл» • «9» – «Дата (месяц и год)»	
votemaxanswernumber	Максимальное количество ответов для вопросов типа Множественный выбор.	Число
votepoints	Количество баллов в шкале. Максимальное значение ранговой шкалы, по которой пользователи будут оценивать варианты предложенных ответов (ранжировать ответы). Используется в вопросах типа “Ранжирование”.	Число
voteshoworder	Порядок вопроса	Число
votecompulsory	Обязательный («true» – вопрос обязателен к заполнению, «false» – нет)	Логическое
votecomment	Комментарий к ответу («true» – пользователь может оставлять комментарии к данному им ответу, «false» – не может)	Логическое
voterangestart	Название начала диапазона для вопросов типа “Выбор из диапазона”.	Строка
voterangeend	Название окончания диапазона для вопросов типа “Выбор из диапазона”.	Строка
voterangeselmethod	Способ выбора ответа для вопросов типа “Выбор из диапазона”: • «0» – Слайдер • «1» – Радиокнопка • «2» – Изображения	Малое число
votescoredquestion	Оцениваемый вопрос («true» – да, «false» – нет). Используется в вопросах типа “Ранжирование”.	Логическое
votecommentcompulsory	Признак обязательности заполнения комментария к ответу: • «true» - обязательно • «false» - не обязательно	Логическое
sursid	Идентификатор раздела вопроса. Заполнен при получении данных опроса, в котором вопросы возвращаются по разделам («surissection» = «true»)	Строка

Структура данных массива вариантов ответа на вопрос **variantAnswer.** По каждой записи массива возвращаются поля:

varid	Идентификатор варианта ответа	Число
vartext	Текст варианта ответа	Строка
varorder	Порядок ответа	Число
varfree	Свободный ответ («true» – есть свободный ответ, «false» – нет)	Логическое
voteidname	Название вопроса	
varscore	Балл за ответ	Число
varrequiredcomment	Заполнение комментария обязательно («true» – да, «false» – нет)	Логическое
varimage	Идентификатор изображения для вопросов типа “Выбор из диапазона” со способом выбора “Изображения”. Получение изображения осуществляется по ссылке: https://URL/mira/Do?doaction=DownloadFileAction&fileId=<fileid>, где fileId - идентификатор изображения.	Число

Структура данных массива сохраненных данных ответов на вопрос **answer.** Массив ответов возвращается по вопросам, где пользователь сохранил ответы, например, для пройденных опросов для отображения сохраненных ответов.

По каждой записи массива возвращаются поля:

Название поля	**Описание**	**Тип**
ansid	Идентификатор записи об ответе	Строка
paid	Идентификатор ответа пользователя	Число

anstext	Текстовый ответ. • Используется для ответов на вопросы типа “Свободный ответ (текст)”. • Для вопросов типа “Одиночный выбор”, “Множественный выбор” в этом поле отображается текстовое название выбранного варианта ответа. • Для вопросов типа “Ранжирование” отображаются текстовые названия ответов в том же порядке, в котором указаны в параметре "varidlist". Названия в виде строки, где каждый ответ разделяется запятой.	Строка
voteid	Идентификатор вопроса, к которому относится ответ	
voteidname	Название вопроса, к которому относится ответ	
varidlist	Идентификаторы выбранных вариантов ответов. Используется для ответов на вопросы типа “Одиночный выбор”, “Множественный выбор”, “Выбор из диапазона”, “Ранжирование”. Для ранжирования – все идентификаторы вариантов ответов в выбранном порядке. Идентификаторы отображаются как строка из значений, разделенных запятыми.	Строка
ansnumber	Числовой ответ. Используется для ответов на вопросы типа “Свободный ответ (число)”.	Число
ansdatetime	Ответ в формате даты и времени. Используется для ответов на вопросы типа “Дата” и “Дата и время”. Формат значения "2013-01-23 02:30:00.000"	Строка
ansdscore	Набранные баллы за вопрос	Дробное число
ansmaxscore	Максимальный балл за вопрос	Число
anscomment	Комментарий к ответу	Строка
files	Массив идентификаторов файлов ответов. Получение файла осуществляется по ссылке: https://URI/mira/Do?doaction=DownloadFileAction&fileId=<fileid>, где fileId - идентификатор файла.	Массив
ansmonth	Месяц. Используется для ответов на вопросы типа “Дата (месяц и год)”. Число от 0 до 11, где 0 – это январь.	Строка
ansyear	Год. Используется для ответов на вопросы типа “Дата (месяц и год)”.	Строка

Информация об ответе на вопрос опроса представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
voteid	Идентификатор voteid вопроса из объекта vote, для которого сохраняется ответ.	Строка
varidlist	Идентификаторы выбранных вариантов ответов. Используется для ответов на вопросы типа “Одиночный выбор”, “Множественный выбор”, “Выбор из диапазона”, “Ранжирование”. Для ранжирования – все идентификаторы вариантов ответов в выбранном порядке.	Массив
text	Текстовый ответ. Используется для ответов на вопросы типа “Свободный ответ (текст)”. Для вопросов типа “Ранжирование” записываются текстовые названия ответов в том же порядке, в котором указаны в параметре "varidlist", в виде строки, где каждый ответ разделяется запятой. Также в параметре text записывается текстовое значение «свободного» варианта ответа для вопросов типа “Одиночный выбор”, “Множественный выбор”. В этом случае при указании текстового значения свободного ответа в поле «text» нужно указать voteid свободного варианта ответа в varidlist.	Строка
number	Числовой ответ. Используется для ответов на вопросы типа “Свободный ответ (число)”. Для пустого значения передается “null”	Число

datetime	Ответ в формате даты и времени. Используется для ответов на вопросы типа “Дата” и “Дата и время”. Формат поля "2020-11-04 00:00:00.000"	Строка
comment	Комментарий к ответу	Строка
files	Загрузка файла осуществляется согласно описанию формата файл	Файл
month	Месяц. Используется для ответов на вопросы типа “Дата (месяц и год)”. Число от 0 до 11, где 0 – это январь. Для пустого значения передается “null”.	Число
year	Год. Используется для ответов на вопросы типа “Дата (месяц и год)”.	Число

Получение списка Моих опросов

Тип запроса: GET

/service/v2/mysurvey

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Набор значений атрибутов для фильтрации – параметр filter

Выходные данные:

- Список назначенных текущему пользователю опросов, с учетом заданного входного фильтра. Выдача результатов – постраничная.

Получение информации об опросе для прохождения

Тип запроса: GET

/service/v2/mysurvey/{id}

Входные параметры:

- Id - Идентификатор опроса пользователя. Список доступных идентификаторов возвращается методом «Получение списка Моих опросов».
- В запрос передаётся авторизация пользователя, полученная по API.

Выходные данные:

- JSON данных опроса по запрошенному id.

Добавление прохождения Моего опроса

Тип запроса: POST

/service/v2/mysurvey/{id}

Входные параметры:

- Идентификатор опроса – Id.
- В запрос передаётся авторизация пользователя, полученная по API.
- В body запроса передается JSON с массивом ответов на вопросы.

Выходные данные:

- http код со статусом сохранения результата – в случае успешного сохранения возвращается http статус 200.

Получение информации об опросе электронного курса по ID для его прохождения

Тип запроса: GET

/service/v2/coursesurvey/{sectionId}

Структура данных опроса аналогична опросу, полученному методом Получение информации об опросе для прохождения.

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Идентификатор раздела-опроса электронного курса sectionId. (Возвращается методом Получение информации о результатах по разделам).

Выходные данные:

- JSON данных опроса с переданным id.

Добавление прохождения опроса электронного курса

Тип запроса: POST

/service/v2/coursesurvey/{sectionId}

Структура данных ответа на опрос аналогична методу Добавление прохождения "Моего опроса".

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Идентификатор раздела электронного курса – sectionId. (Возвращается методом Получение информации о результатах по разделам).
- В body запроса передается JSON с массивом ответов на вопросы.

Выходные данные:

- http код со статусом сохранения результата – в случае успешного сохранения возвращается http статус 200.

Модуль reportTemplates (Шаблоны отчетов)

Информация о шаблоне отчета представляется следующей моделью данных:

Название поля	Описание	Тип
rtid	Идентификатор шаблона отчета	Число
rtname	Название шаблона отчета	Строка(255)
fileid	Идентификатор файла шаблона отчета	Файл
rtdescription	Описание шаблона отчета	Строка

Добавление шаблона отчета

Тип запроса: POST

/service/v2/reportTemplates

Входные параметры:

- Набор из параметров модели шаблона отчета.

Выходные данные:

- Идентификатор созданного шаблона отчета.

Получение информации о шаблоне отчета

Тип запроса: GET

/service/v2/reportTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор шаблона отчета.

Выходные данные:

- Параметры из модели шаблона отчета.

Изменение информации о шаблоне отчета

Тип запроса: PUT

/service/v2/reportTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор шаблона отчета.

Входные параметры:

- Набор из параметров модели шаблона отчета, которые требуется изменить.

Удаление шаблона отчета

Тип запроса: DELETE

/service/v2/reportTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого шаблона отчета.

Построение отчета с возвращением данных в JSON

Тип запроса: GET

/service/v2/reportTemplates/{id}/report/json

Параметр URL (обязательный):

- Идентификатор шаблона отчета.

Входные параметры:

- Системные названия параметров отчета.

На заметку

– *Системные названия параметров отчета отображаются на карточке **шаблона отчета** на вкладке **Просмотр отчета** в скобках.*

– *Если отчет не содержит параметров, параметры не указываются.*

Выходные данные:

- Массив объектов, состоящих из значений «номер столбца отчета»: «данные столбца отчета».
    - Количество значений в каждом объекте соответствует количеству столбцов в отчете.
    - Столбцы нумеруются с нуля.
    - В первом значении массива возвращается сопоставление номеров и названий столбцов отчета.

Например

```
[{
   "1": "Участник",
   "0": "Мероприятие"
}, {
   "1": " Иванов Иван Иванович",
   "0": "Вебинар"
}, {
   "1": "Петров Петр Петрович",
   "0": "Тестирование"
}]
```

Построение отчета с возвращением данных в XML

Тип запроса: GET

/service/v2/reportTemplates/{id}/report/xml

Параметр URL (обязательный):

- Идентификатор шаблона отчета.

Входные параметры:

- Системные названия параметров отчета.

На заметку

– *Системные названия параметров отчета отображаются на карточке **шаблона отчета** на вкладке **Просмотр отчета** в скобках.*

– *Если отчет не содержит параметров, параметры не указываются.*

Выходные данные:

- Корневой элемент <report> содержащий элементы <row>. Каждый элемент <row> состоит из элементов <column> с атрибутом «num», содержащим номер столбца. Значение столбца передается в значении элемента column.
    - Количество элементов <column> в каждом элементе <row> соответствует количеству столбцов в отчете.
    - Столбцы нумеруются с нуля.
    - В первом значении <row> возвращается сопоставление номеров и названий столбцов отчета.

Например

```
<report>
```

```
<row>
<column num="0">Мероприятие</column>
<column num="1">Участник</column>
</row>
<row>
<column num="0">Вебинар</column>
<column num=»1»>Иванов Иван Иванович</column>
</row>
<row>
<column num="0">Тестирование</column>
<column num=»1»>Петров Петр Петрович</column>
</row>
</report>
```

Модуль reports (Отчеты)

Информация об отчете представляется следующей моделью данных:

Название поля	Описание	Тип
reportid	Идентификатор отчета	Число
reportname	Название отчета	Строка(255)
rtid	Идентификатор шаблона отчета	Число
reportfilename	Название файла отчета	Строка
reportdescription	Описание отчета	Строка
reportissend	Рассылать	Малое
reportmailsubject	Тема письма	Строка(100)
reportaddemaillist	Внесистемные адресаты	Строка(255)
reportmailtemplate	Шаблон письма	Строка(2000)

Добавление отчета

Тип запроса: POST

/service/v2/reports

Входные параметры:

- Набор из параметров модели отчета.

Выходные данные:

- Идентификатор созданного отчета.

Получение информации об отчете

Тип запроса: GET

/service/v2/reports/{id}

Параметр URL (обязательный):

- id – идентификатор отчета.

Выходные данные:

- Параметры из модели отчета.

Изменение информации об отчете

Тип запроса: PUT

/service/v2/reports/{id}

Параметр URL (обязательный):

- id – идентификатор отчета.

Входные параметры:

- Набор из параметров модели отчета, которые требуется изменить.

Удаление отчета

Тип запроса: DELETE

/service/v2/reports/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого отчета.

Модуль certificates (Выданные сертификаты)

Информация о сертификатах представляется следующей моделью данных:

Название поля	Описание	Тип
certid	Id выданного сертификата	Число
personid	Id физического лица, которому выдан сертификат	Число
certcreatedate	Дата выдачи сертификата	Дата

Получение информации сертификате

Тип запроса: GET

/service/v2/certificates/{id}

Параметр URL (обязательный):

- ID выданного сертификата.

Выходные данные:

- Параметры из модели сертификата.

Изменение информации о выданном сертификате

Тип запроса: PUT

/service/v2/certificates/{id}

Параметр URL (обязательный):

- ID выданного сертификата.

Входные параметры:

- Набор из параметров модели сертификата, которые требуется изменить.

Удаление выданного сертификата

Тип запроса: DELETE

/service/v2/certificates/{id}

Параметр URL (обязательный):

- ID выданного сертификата.

Получение списка сертификатов пользователя

Информация о сертификате пользователя представляется следующей моделью данных:

Название поля	Описание	Тип
certid	Идентификатор выданного сертификата	Число
certidname	Название выданного сертификата	Строка
certtypeid	Id шаблона сертификата	Число
certtypeidname	Название шаблона сертификата	Строка
personid	Id физического лица, которому выдан сертификат	Число
personidname	ФИО физического лица, которому выдан сертификат	Строка
certcreatedate	Дата выдачи сертификата	Дата
meid	Id мероприятия	Число
meidname	Название мероприятия	Строка
certcreatedate	Дата выдачи сертификата	Дата
link	Ссылка на скачивание сертификата. Для скачивания сертификата необходима авторизация в системе под пользователем, которому выдан сертификат. Через API сертификат возможно скачать, авторизовавшись с помощью модуля auth. Ссылка возвращается только для выданных сертификатов, по котором доступно скачивание электронной версии.	Строка

Тип запроса: GET

/service/v2/certificates/byPerson/{id}

Параметр URL (обязательный):

- Идентификатор физического лица.

Выходные данные:

- Массив объектов сертификата пользователя.

Модуль certTypes (Шаблоны сертификатов)

Получение списка шаблонов сертификатов по мероприятиям пользователя

Метод возвращает список шаблонов сертификатов, по которым пользователь может получить сертификат в результате прохождения мероприятий. В списке возвращаются записи о шаблонах сертификатов по мероприятиям, по которым у пользователя:

- Статус доступа для прохождения – «Открыт»;

- Статус завершения обучения - «Не завершено»;
- Не получен сертификат по мероприятию и шаблону сертификата.

Информация о шаблоне сертификата представляется следующей моделью данных:

Название поля	Описание	Тип
certtypeid	Идентификатор шаблона сертификата	Число
certtypeidname	Название шаблона сертификата	Строка
meid	Идентификатор мероприятия	Число
meidname	Название мероприятия	Строка
certtypescore	Минимальная оценка в процентах для получения сертификата	Число
certtypemaxscore	Максимальная оценка в процентах для получения сертификата	Число
certtypememberstatus	Статус (0 - Не пройден, 1 - Пройден, 3 - В процессе, 4 - Не начат, 5 - На проверке, 6 - Проверено, 7 - Не важно)	Малое

Тип запроса: GET

/service/v2/certTypes/byPerson/{id}

Параметр URL (обязательный):

- Идентификатор физического лица.

Выходные данные:

- Массив объектов шаблонов сертификатов.

Модуль mediaResources (Ресурсы медиатеки)

Информация о ресурсе представляется следующей моделью данных:

Название поля	Описание	Тип
rid	Идентификатор ресурса	Число
rname	Название ресурса	Строка(255)
rdesc	Описание ресурса	Строка(4000)
rcreatedate	Дата создания ресурса	Дата
rchangedate	Дата изменения ресурса	Дата
rkeywords	Ключевые слова	Строка(255)
fileid	Идентификатор файла (только для ресурса вида Файл). Для скачивания файла необходимо сформировать ссылку вида: https://test.lmsonline.ru/mira/Do?doaction=DownloadFileAction&fileId=<fileid>, заменив <fileid> на присланное значение в fileid и добавив параметр usrsesid с идентификатором сессии, полученном через модуль auth.	Файл
rurl	Ссылка на внешний ресурс (только для ресурса вида Внешний ресурс)	Строка(1000)
rexternaldata	Код для вставки (только для ресурса вида Встраиваемый внешний ресурс)	Строка
personid	Идентификатор пользователя, создавшего ресурс	Число
personidname	ФИО пользователя, создавшего ресурс	Строка
rkind	Вид ресурса (0 - Файл, 1 - Wiki, 2 - Ссылка на внешний ресурс, 3 - Встраиваемый внешний ресурс)	Малое
risdownload	Доступ к файлу (0 - не доступен, 1 - доступен)	Малое
statusrsid	Идентификатор статуса ресурса	Число
statusrsidname	Название статуса ресурса	Строка (100)
checkpersonid	Идентификатор пользователя, изменившего статус	Число
checkpersonidname	ФИО пользователя, изменившего статус	Строка
rpreview	Предпросмотр (0 - Авто, 1 – Из файлов предпросмотра, 2 - Выключен)	Малое
riscomment	Разрешить оценивание и комментарии	Логическое
contentid	Идентификатор контента	Число
contentidname	Название контента	Строка
rversionaccessedit	Управление версиями (0 - Все, 1 - Модераторы, 2 - Авторы и модераторы)	Малое

rversionaccessview	Доступ к просмотру версий (0 - Все, 1 - Модераторы, 2 - Авторы и модераторы)	Малое
rrelevancedate	Дата актуальности	Дата
rstatisticenabled	Включена ли статистика (0 – Нет, 1 – Да)	Логическое
rchangeauthor	Идентификатор пользователя, изменившего текущую версию	Число
rchangeauthorname	ФИО пользователя, изменившего текущую версию	Строка
donotconvert	Не конвертировать	Логическое
rallowprinting	Разрешить печать	Логическое

Добавление ресурса

Тип запроса: POST

/service/v2/mediaResources

Входные параметры:

- Набор из параметров модели ресурса. Обязательным параметром является rname.

Выходные данные:

- Параметры из модели созданного ресурса.

Получение информации о ресурсе

Тип запроса: GET

/service/v2/mediaResources/{id}

Параметр URL (обязательный):

- id – идентификатор ресурса.

Выходные данные:

- Параметры из модели ресурса.

Изменение информации о ресурсе

Тип запроса: PUT

/service/v2/mediaResources/{id}

Параметр URL (обязательный):

- id – идентификатор ресурса.

Входные параметры:

- Набор из параметров модели ресурса, которые требуется изменить.

Выходные данные:

- Параметры из модели ресурса

Удаление ресурса

Тип запроса: DELETE

/service/v2/mediaResources/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого ресурса.

Модуль mediaUserGroups (Пользовательские группы ресурсов)

Информация о пользовательских группах ресурсов представляется следующей моделью данных:

Название поля	Описание	Тип
ugrid	Идентификатор пользовательской группы	Число
ugrname	Название пользовательской группы	Строка
ugrparentid	Идентификатор родительской пользовательской группы. Обязательное поле. Если нужно разместить группу «в корне» пользовательских групп, нужно указать значение «-10», соответствующее системной пользовательской группе «ПКР».	Число
ugrparentidname	Название родительской пользовательской группы	Строка
ugrdesc	Стандартное описание пользовательской группы	Строка
ugrextdesc	Расширенное описание пользовательской группы	Строка
personid	Идентификатор создателя пользовательской группы	Число
personidname	ФИО создателя пользовательской группы	Строка
ugrcreatedate	Дата создания пользовательской группы	Дата
ugrshowdesc	Показывать описания группы (0 – Не показывать, 1 – Показывать стандартное, 2 – Показывать расширенное)	Малое
ugrshowsubgroup	Показывать подгруппы (0 – Не показывать, 1 – Первого уровня, 2 – Всех подуровней, 3 - заданное число, 4 – Показывать подгруппы как списки с категориями, 5 – Показать, как список, 6 – Показать подгруппы как списки без категорий)	Малое
ugrshowsubgroupimage	Показывать изображения подгрупп	Логическое
ugrshowsubgroupdesc	Показывать описания подгрупп (0 – Не показывать, 1 – Показывать стандартные, 2 – Показывать расширенные)	Малое
ugrsubgrouplevelcount	Количество показываемых подуровней	Число
ugrorder	Порядок	Число
ugrcolumn	Количество колонок	Число
ugrsortorder	Сортировать элементы (0 – По алфавиту, 1 – По дате добавления)	Малое
ugrsortdirection	Порядок сортировки (0 – По возрастанию, 1 – По убыванию)	Малое
objcount	Количество объектов в пользовательской группе (дочерние пользовательские группы не учитываются)	Число

Добавление пользовательской группы

Тип запроса: POST

/service/v2/mediaUserGroups

Входные параметры:

- Набор из параметров модели пользовательской группы. Обязательным параметром является название и родительская пользовательская группа.

Выходные данные:

- Параметры из модели созданной пользовательской группы.

Получение информации о пользовательской группе

Тип запроса: GET

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Параметры из модели пользовательской группы.

Изменение информации о пользовательской группе

Тип запроса: PUT

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Входные параметры:

- Набор из параметров модели пользовательской группы, которые требуется изменить.

Выходные данные:

- Параметры из модели пользовательской группы.

Удаление пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой пользовательской группы.

Добавление дочерней пользовательской группы

Тип запроса: POST

/service/v2/mediaUserGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор добавляемой дочерней пользовательской группы.

Исключение дочерней пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.

- childId – идентификатор исключаемой дочерней пользовательской группы.

Получение списка дочерних пользовательских групп

Тип запроса: GET

/service/v2/mediaUserGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Массив объектов с параметрами из модели пользовательской группы.

Добавление ресурса в пользовательскую группу

Тип запроса: POST

/service/v2/mediaUserGroups/{id}/childObjects/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор добавляемого ресурса.

Исключение ресурса из пользовательской группы

Тип запроса: DELETE

/service/v2/mediaUserGroups/{id}/childObjects/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор исключаемого ресурса.

Получение списка ресурсов пользовательской группы

Тип запроса: GET

/service/v2/mediaUserGroups/{id}/childObjects

Параметр URL (обязательный):

- id – идентификатор пользовательской группы.

Выходные данные:

- Массив объектов модели ресурса.

Модуль mediaGroups (Группы ресурсов)

Информация о группах ресурсов представляется следующей моделью данных:

Название поля	Описание	Тип

gcid	Идентификатор группы	Число
gcname	Название группы	Строка(100)
gcdesc	Описание группы	Строка(255)
gcparentid	Идентификатор родительской группы	Число
gcparentidname	Название родительской группы	Строка(100)
grkind	Вид группы (0 – Динамическая, 1 – Полудинамическая, 2 – Статическая);	Малое

Добавление группы

Тип запроса: POST

/service/v2/mediaGroups

Входные параметры:

- Набор из параметров модели группы. Обязательным параметром является название.

Выходные данные:

- Параметры из модели созданной пользовательской группы.

Получение информации о группе

Тип запроса: GET

/service/v2/mediaGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Параметры из модели группы.

Изменение информации о группе

Тип запроса: PUT

/service/v2/mediaUserGroups/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Входные параметры:

- Набор из параметров модели группы, которые требуется изменить.

Выходные данные:

- Параметры из модели группы

Удаление группы

Тип запроса: DELETE

/service/v2/mediaGroups/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой группы.

Добавление дочерней группы

Тип запроса: POST

/service/v2/mediaGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор пользовательской группы.
- childId – идентификатор добавляемой дочерней пользовательской группы.

Исключение дочерней группы

Тип запроса: DELETE

/service/v2/mediaGroups/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемой дочерней группы.

Получение списка дочерних групп

Тип запроса: GET

/service/v2/mediaGroups/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами из модели группы.

Добавление ресурса в группу

Тип запроса: POST

/service/v2/mediaGroups/{id}/childMedia/{childId}

Параметр URL (обязательный):

- id – идентификатор группы.
- childId – идентификатор добавляемого ресурса.

Исключение ресурса из группы

Тип запроса: DELETE

/service/v2/mediaGroups/{id}/childMedia/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемого ресурса.

Получение списка ресурсов группы

Тип запроса: GET

/service/v2/mediaGroups/{id}/childMedia

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов модели ресурса.

Модуль favorites (Избранное)

Информация об избранном представляется следующей моделью данных:

Название поля	Описание	Тип
objtablename	Тип избранного объекта. Поддерживаемый тип – **measure** (мероприятие).	Строка
objtableid	Идентификатор избранного объекта	Число
personid	Идентификатор физического лица, добавившего объект в избранное	Число
personidname	ФИО физического лица, добавившего объект в избранное	Строка

Добавление в избранное

Тип запроса: POST

service/v2/favorites/{personId}/{objtablename}/{objectId}

Параметры URL:

- personId – идентификатор физического лица, для которого добавляется запись избранного.
- objtablename – тип объекта, добавляемого в избранное.
- objectId – id объекта, добавляемого в избранное.

Выходные данные:

- Параметры из модели созданного избранного.

Например

https://test.lmsonline.ru/mira/service/v2/favorites/14/measure/340?appid=system&sign=B183A48E0B7E9C2AFD101A2EC7AA5054

Получение избранного пользователя

Тип запроса: GET

/service/v2/favorites/{personId}

Параметры URL:
- personId – идентификатор физического лица.

Выходные данные:
- Массив записей с параметрами модели избранного. Данные возвращаются постранично.

Получение избранного пользователя по определенному объекту

Тип запроса: GET

/service/v2/favorites/{personId}/{objtablename}

Параметры URL:
- personId – идентификатор физического лица.
- objtablename – тип сущности.

Выходные данные (данные возвращаются постранично):
- objtableid – идентификатор объекта.
- objectname – название объекта.

Удаление записи избранного

Тип запроса: DELETE

/service/v2/favorites/{personId}/{objtablename}/{objectId}

Параметры URL (обязательные):
- personId – идентификатор физического лица, для которого удаляется запись об избранном.
- objtablename – тип объекта по которому удаляется запись избранного.
- objectId – id объекта по которому удаляется запись избранного.

Модуль tests (Тестирование)

Модуль предназначен для получения данных теста для прохождения тестирования во внешнем плеере. Метод Получение данных тестирования должен вызываться промежуточным сервисом, который получит данные системы и передаст на клиентскую часть для тестирования пользователя. Нужно учитывать, что данные теста могут различаться при разных запусках теста. Например, если в тесте используется случайная выборка 10 вопросов из 20, то при каждом вызове метода будет возвращаться 10 случайно выбранных вопросов. Данные об ответах и результатах пользователя должны быть переданы клиентской частью обратно промежуточному сервису, который сохранит результат прохождения в системе по алгоритму:

1) При запуске теста пользователем нужно создать попытку прохождения методом Добавление попытки с параметрами csid = testcsid (параметр, получаемый при запросе информации о мероприятиях участника либо при запросе списка мероприятий авторизованного пользователя для мероприятий типа Электронный тест) и mmid. Будет создана попытка прохождения и возвращен идентификатор попытки attid.
2) Ответ на каждый вопрос нужно сохранять методом Добавление ответа участника на вопрос.
3) При завершении прохождения тестирования нужно обновить данные по созданной попытке прохождения теста методом Изменение попытки.

Поддерживаются вопросы типов:

- Одиночный выбор;
- Множественный выбор;
- Соответствие;
- Порядок.

Данные тестирования представлены следующей моделью данных:

Название поля	**Описание**	**Тип**
Настройки, соответствующие блоку настроек теста Основная информация в интерфейсе		
name	Название теста	Строка
duration	Время на тест (сек)	Число
isShowFeedback	Показывать окно обратной связи	Логическое
Блок настроек теста Начальная страница		
isShowTestPage	Показывать начальную страницу теста	Логическое
comment	Текст для начальной страницы теста	Строка
Настройки, соответствующие блоку настроек теста Расчет результата теста в интерфейсе		
resultSettings	Расчет результата теста (0 - Стандартный, 1 - По формуле, 2 - По весу)	Малое
formula	Формула	Строка
score	Условия успешной сдачи	Число
resultCondition	Баллы/Проценты (0 - Баллы, 1 - Проценты)	Малое
procent	Минимальный процент вопросов, на которые должен быть дан ответ	Число
Настройки, соответствующие блоку настроек теста Показ результатов в интерфейсе		
isShowResult	Показывать блок результатов	Логическое
showResultColumnQuestion	Показывать название вопроса (колонка Вопрос)	Логическое
showResultColumnResult	Показывать результат по вопросам (колонка Результат)	Логическое
allowReview	Показывать результаты по вопросам (колонка Баллы)	Логическое
showResultColumnUserAnswer	Показывать ответы по вопросам (колонка Ваш ответ)	Логическое
isShowRightAnswerRes	Показывать правильные ответы (колонка Правильный ответ)	Логическое
isShowTotalScore	Показывать общее количество баллов	Логическое
isShowResultStatus	Показывать статус прохождения	Логическое
isShowSuccesMessage	Показывать сообщение при удачном завершении теста	Логическое
successMessage	Сообщение при удачном завершении	Строка
isShowUnsuccessMessage	Показывать сообщение при неудачном завершении теста	Логическое
unsuccessMessage	Сообщение при неудачном завершении	Строка
isShowResComment	Показывать комментарии (интерпретацию) по результатам разделов	Логическое
finishMessage	Сообщение на последней странице тестирования	Строка
Настройки, соответствующие блоку настроек теста Настройки оглавления в интерфейсе		
isShowQType	Показывать тип вопроса	Логическое
isShowView	Показывать просмотр вопроса	Логическое
isShowAnswer	Показывать наличие ответа	Логическое
isShowScore	Показывать набранные баллы	Логическое
allowQuestionGo	Разрешить переходы к вопросам	Логическое
Настройки, соответствующие блоку настроек теста Показ вопросов и ответов в интерфейсе		
viewQuestion	Показ вопросов («BY_ONE» - По одному вопросу, «BY_SECTION» - По разделам, «ALL» - Все вопросы)	Строка
isShowQName	Показывать названия вопросов	Логическое
isShowInstruction	Показывать инструкции к вопросам	Логическое
isShowRightAnswer	Показывать правильные ответы после вопросов	Логическое
isShowAnswerComment	Показывать комментарии к ответам	Логическое
isShowRightMessage	Показывать сообщения для правильных ответов	Логическое
rightMessage	Общее для теста сообщение при правильном ответе	Строка
isShowErrorMessage	Показывать сообщения для неправильных ответов	Логическое
errorMessage	Общее для теста сообщение при неправильном ответе	Строка
isShowAnswerAfterQuestion	Показывать ответ после вопроса	Логическое
sections	Массив данных по разделам теста (данные передаются по каждому разделу теста)	Массив

Структура массива данных настроек раздела теста sections:

Название поля	Описание	Тип
name	Название раздела	Строка
duration	Время на раздел (сек)	Число
score	Проходной балл	Число (дробное)
unsuccessMessage	Сообщение при неудачном завершении	Текст
isShowStartPage	Показывать начальную страницу раздела	Текст
comment	Комментарий к разделу	Текст
attemptCount	Количество попыток на вопрос	Число
commentSettings	В чем заданы данные в массиве Комментарии (интерпретация) к результатам для раздела теста (0 – В баллах, 1- В процентах)	Малое
resultComments	Массив данных комментариев к результатам раздела теста (для каждого раздела теста)	Массив
questions	Массив вопросов раздела (для каждого раздела теста)	Массив

Структура массива данных комментариев к результатам теста resultComments:

Название поля	Описание	Тип
minScore	От	Число
maxScore	До	Число
Comment	Комментарий	Строка

Структура массива данных настроек вопроса questions:

Название поля	Описание	Тип
qid	Идентификатор вопроса (используется для проставления результатов)	Число
header	Название вопроса	Строка
type	Тип (Одиночный выбор – «single», Множественный выбор – «multiple», Соответствие – «match», Порядок – «order»)	Строка
text	Текст вопроса	Строка
score	Количество баллов за вопрос	Число
byWeight	Считать баллы по ответам	Малое
attemptCount	Количество попыток на вопрос	Число
Matches	Массив соответствий (по каждому вопросу типа Соответствие)	Массив
Answers	Массив ответов вопроса (по каждому вопросу)	Массив

Структура массива данных соответствий Matches:

Название поля	Описание	Тип
identifier	Идентификатор соответствия	Число
Text	Текст	Строка
index	Порядок	Число

Структура массива данных ответов вопроса Answers:

Название поля	Описание	Тип
Identifier	Идентификатор ответа (используется для проставления результатов)	Число
text	Текст	Строка
weight	Балл	Число
correct	Правильный (для вопросов типов Одиночный выбор, Множественный выбор). 0 - Неправильный, 1 - Правильный, 2 – Нейтральный	Число
order	Правильный порядок (для ответа на вопрос типа Порядок)	Число
rightMatches	Массив соответствий, правильных для ответа (по каждому вопросу типа Соответствие)	Массив

Структура массива данных соответствий rightMatches:

Название поля	Описание	Тип
Identifier	Идентификатор соответствия	Число
Text	Текст соответствия	Строка
Index	Порядок	Число

Получение данных тестирования

Тип запроса: GET

service/v2/tests/{testid}/user

Параметры URL:

- testid– идентификатор теста. Идентификатор теста возвращается запросом Получение списка мероприятий авторизованного пользователя или запросом Получении списка мероприятий пользователя в качестве участника.

Входные параметры:

- usrsesid – id сессии, возвращаемый методом Открытие пользовательской сессии.

Выходные данные:

- Данные теста.

Модуль messages (Личные сообщения)

Информация о теме личного сообщения представляется следующей моделью данных:

Название поля	Описание	Тип
topicid	Идентификатор темы	Число
topicname	Название темы	Строка
topiccreatedate	Дата создания темы (если параметр не указан при создании сообщения, заполняется автоматически)	Дата
creatorid	Идентификатор физического лица, создавшего тему	Число
creatoridname	ФИО физического лица, создавшего тему (используется только при получении данных)	Строка
addressid	Идентификатор физического лица – адресата темы	Число
addressidname	ФИО физического лица – адресата темы (используется только при получении данных)	Строка
isread	Прочитана ли тема	Логическое
lastpersonid	Идентификатор последнего прочитавшего тему физического лица (используется только при получении данных)	Число
lastpersonidname	ФИО последнего прочитавшего тему физического лица (используется только при получении данных)	Строка
topicupdatedate	Дата последнего изменения темы	Дата
isreadbyauthor	Прочитано ли изменение темы автором	Логическое

Информация о сообщении представляется следующей моделью данных:

Название поля	Описание	Тип
msid	Идентификатор сообщения	Число
mstext	Текст сообщения (поле обязательно должно иметь значение)	Строка
createdate	Дата создания сообщения (если параметр не указан при создании сообщения, заполняется автоматически)	Дата
creatorid	Идентификатор физического лица, создавшего сообщение	Число
creatoridname	ФИО физического лица, создавшего сообщение (используется только при получении данных)	Строка
topicid	Идентификатор темы	Число
topicidname	Название темы (используется только при получении данных)	Строка

addressid	Адресат сообщения	Число
addressidname	ФИО физического лица – адресата сообщения (используется только при получении данных)	Строка

Добавление темы

Тип запроса: POST

/service/v2/messages

Входные параметры:

- Набор из параметров модели темы, обязательные – topicname, creatorid и addressid.
- mstext – текст первого сообщения в теме.

Выходные данные:

- Идентификатор созданной темы.

Получение информации о теме

Тип запроса: GET

/service/v2/messages/{id}

Параметр URL (обязательный):

- Идентификатор темы

Выходные данные:

- Параметры из модели темы

Изменение информации о теме

Тип запроса: PUT

/service/v2/messages/{id}

Параметр URL (обязательный):

- id – идентификатор темы.

Входные параметры:

- Набор из параметров модели темы, которые требуется изменить.

Выходные данные:

- Параметры из модели измененной темы

Удаление темы

Тип запроса: DELETE

/service/v2/messages/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой темы.

Добавление сообщения

Тип запроса: POST

/service/v2/messages/{topicid}/messages

Параметр URL (обязательный):

- Идентификатор темы, в которую добавляется сообщение.

Входные параметры:

- Набор из параметров модели сообщения, обязательный - mstext.

Выходные данные:

- Идентификатор созданного сообщения.

Получение информации о сообщении

Тип запроса: GET

/service/v2/messages/messages/{id}

Параметр URL (обязательный):

- Идентификатор сообщения.

Выходные данные:

- Параметры из модели сообщения.

Изменение сообщения

Тип запроса: PUT

/service/v2/messages/messages/{id}

Параметр URL (обязательный):

- id – идентификатор сообщения.

Входные параметры:

- Набор из параметров модели сообщения, которые требуется изменить.

Выходные данные:

- Параметры из модели измененного сообщения.

Удаление сообщения

Тип запроса: DELETE

/service/v2/messages/messages/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого сообщения.

На заметку

При удалении последнего сообщения в теме, тема также удаляется.

Получение списка сообщений темы

Тип запроса: GET

/service/v2/messages/{topicid}/messages

Параметр URL (обязательный):

- Идентификатор темы.

Выходные данные:

- Массив объектов с параметрами сообщения. Данные возвращаются постранично.

Модуль comments (Комментарии)

Информация о комментарии представляется следующей моделью данных:

Название поля	Описание	Тип
objtableid	Идентификатор объекта, к которому относится комментарий	Число
objtablename	Системное название комментируемого объекта. Возможные значения: • «blogrecord» - запись блога; • «cmnote» – объявление; • «measure» – мероприятие; • «media» – ресурс; • «publication» – публикация; • «quote» – опрос; • «event» - событие; • «queryrequest» - заявка.	Строка
personid	Идентификатор физического лица, создавшего комментарий	Число
personidname	ФИО физического лица, создавшего комментарий (используется только при получении данных)	Строка
cdate	Дата создания комментария	Дата
ctext	Текст комментария	Строка

Добавление комментария

Тип запроса: POST

/service/v2/comments

Входные параметры:

- Набор из параметров модели комментария. Обязательные – objtableid, objtablename, personid, ctext.

Выходные данные:

- Идентификатор созданного комментария.

Получение информации о комментарии

Тип запроса: GET

/service/v2/comments/{id}

Параметр URL (обязательный):

- id – идентификатор комментария.

Выходные данные:

- Параметры из модели комментария.

Изменение информации о комментарии

Тип запроса: PUT

/service/v2/comments/{id}

Параметр URL (обязательный):

- id – идентификатор комментария.

Входные параметры:

- Набор из параметров модели комментария, которые требуется изменить.

Удаление комментария

Тип запроса: DELETE

/service/v2/comments/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого комментария.

Модуль galleries (Галереи)

Информация о галерее представляется следующей моделью данных:

Название поля	Описание	Тип
glid	Идентификатор галереи	Число
glname	Название галереи	Строка
gldesc	Описание галереи	Строка

Добавление галереи

Тип запроса: POST

/service/v2/galleries

Входные параметры:

- Набор из параметров модели галереи. Обязательный – glname.

Выходные данные:

- Идентификатор созданной галереи.

Получение информации о галерее

Тип запроса: GET

/service/v2/galleries/{id}

Параметр URL (обязательный):

- id – идентификатор галереи.

Выходные данные:

- Параметры из модели галереи.

Изменение информации о галерее

Тип запроса: PUT

/service/v2/galleries/{id}

Параметр URL (обязательный):

- id – идентификатор галереи.

Входные параметры:

- Набор из параметров модели галереи, которые требуется изменить.

Удаление галереи

Тип запроса: DELETE

/service/v2/galleries/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой галереи.

Модуль importTemplates (Шаблоны импорта)

Информация о шаблоне импорта представляется следующей моделью данных:

Название поля	Описание	Тип
itid	Идентификатор шаблона импорта	Число
itname	Название шаблона импорта	Строка
itformat	Формат шаблона импорта (0 - Xls, 1 - Xml)	Число

Добавление шаблона импорта

Тип запроса: POST

/service/v2/importTemplates

Входные параметры:

- Набор из параметров модели шаблона импорта. Обязательный – itname и itformat.

Выходные данные:

- Идентификатор созданного шаблона импорта.

Получение информации о шаблоне импорта

Тип запроса: GET

/service/v2/importTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор шаблона импорта.

Выходные данные:

- Параметры из модели шаблона импорта.

Изменение информации о шаблоне импорта

Тип запроса: PUT

/service/v2/importTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор шаблона импорта.

Входные параметры:

- Набор из параметров модели шаблона импорта, которые требуется изменить.

Удаление шаблона импорта

Тип запроса: DELETE

/service/v2/importTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого шаблона импорта.

Модуль importTasks (Задачи импорта)

Информация о задаче импорта представляется следующей моделью данных:

Название поля	Описание	Тип
itid	Идентификатор задачи импорта	Число
itname	Название задачи импорта	Строка
itstatus	Статус задачи импорта (0 - Данные не загружены, 1 - Данные загружаются, 2 - Ошибки в данных, 3 - Данные загружены, 4 - Данные импортируются, 5 - Данные импортированы, 6 - Данные импортированы с ошибками, 7 - Системная ошибка)	Малое
itfiles	Массив идентификаторов файлов импорта. Возможно добавление файлов по API.	Массив чисел

Информация о правиле импорта представляется следующей моделью данных:

Название поля	Описание	Тип

ittid	Идентификатор правила импорта	Число
ittname	Название правила импорта	Строка
taskid	Идентификатор задачи импорта	Число
templateid	Идентификатор шаблона импорта	Число
ittsearchtype	Тип поиска (0 - Любой файл, 1 - Имя файла содержит, 2 - Содержимое файла содержит)	Число
ittsearchregexp	Текст регулярного выражения для поиска	Строка

Добавление задачи импорта

Тип запроса: POST

/service/v2/importTasks

Входные параметры:

- Набор из параметров модели задачи импорта. Обязательные – itname и itfiles.

Выходные данные:

- Идентификатор созданной задачи импорта.

Получение информации о задаче импорта

Тип запроса: GET

/service/v2/importTasks/{id}

Параметр URL (обязательный):

- id – идентификатор задачи импорта.

Выходные данные:

- Параметры из модели задачи импорта.

Изменение информации о задаче импорта

Тип запроса: PUT

/service/v2/importTasks/{id}

Параметр URL (обязательный):

- id – идентификатор задачи импорта.

Входные параметры:

- Набор из параметров задачи импорта, которые требуется изменить.

Удаление задачи импорта

Тип запроса: DELETE

/service/v2/importTasks/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой задачи импорта.

Добавление правила импорта

Тип запроса: POST

/service/v2/importTasks/{taskId}/taskTemplates

Параметр URL (обязательный):

- taskId – идентификатор задачи импорта, для которой добавляется правило импорта.

Входные параметры:

- Набор из параметров модели правила импорта. Обязательные – ittname, templateid и ittsearchtype.

Выходные данные:

- Идентификатор созданного правила импорта.

Получение информации о правиле импорта

Тип запроса: GET

/service/v2/importTasks/taskTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор задачи правила.

Выходные данные:

- Параметры из модели правила импорта.

Изменение информации о правиле импорта

Тип запроса: PUT

/service/v2/importTasks/taskTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор правила импорта.

Входные параметры:

- Набор из параметров задачи импорта, которые требуется изменить.

Удаление правила импорта

Тип запроса: DELETE

/service/v2/importTasks/taskTemplates/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого правила импорта.

Получение списка правил импорта задачи импорта

Тип запроса: GET

/service/v2/importTasks/{taskId}/taskTemplates

Параметр URL (обязательный):

- taskId – идентификатор задачи импорта.

Выходные данные:

- Массив объектов с параметрами правил импорта. Данные возвращаются постранично.

Загрузка данных задачи импорта из файла

Тип запроса: PUT

/service/v2/importTasks/{taskId}/load

Параметр URL (обязательный):

- taskId – идентификатор задачи импорта.

Запуск задачи импорта данных

Тип запроса: PUT

/service/v2/importTasks/{taskId}/import

Параметр URL (обязательный):

- taskId – идентификатор задачи импорта.

Модуль myProfile (Мой профиль)

Модуль позволяет сохранять и получать по API данные, связанные с текущим пользователем, с помощью авторизации по API без подписи запроса, токен авторизации пользователя является обязательным при вызове каждого метода. Также, модуль предоставляет возможность связывать с текущим пользователем произвольные объекты в текстовом формате.

Информация об объектах, связанных с текущим пользователем представляется следующей моделью данных:

Название поля	Описание	Тип
id	Идентификатор объекта	Число
no	Номер	Строка (100)
type	Тип	Строка (255)
name	Название	Строка (1000)
text	Текст	Строка (10000)

Получение информации о текущем пользователе

Тип запроса: GET

/service/v2/myProfile

Входные параметры:

- usrsesid – идентификатор сессии пользователя, полученный с помощью авторизации по API (обязательный параметр).

Выходные данные:

- Данные модели физического лица.

Получение информации о группах текущего пользователя

Тип запроса: GET

/service/v2/myProfile/groups

Входные параметры:

- usrsesid – идентификатор сессии пользователя, полученный с помощью авторизации по API (обязательный параметр).

Выходные данные:

- Массив объектов с параметрами модели группы физических лиц. Данные возвращаются постранично.

Добавление произвольного объекта для пользователя

Тип запроса: POST

/service/v2/myProfile/objects

Входные параметры:

- Набор из параметров модели объекта пользователя.
- usrsesid – идентификатор сессии пользователя, полученный с помощью авторизации по API (обязательный параметр).

Выходные данные:

- Параметры созданного объекта.

Получение информации об объектах пользователя

Тип запроса: GET

/service/v2/myProfile/objects

Входные параметры:

- Параметры модели объекта пользователя. При указании какого-либо параметра со значением производится фильтрация данных по этому параметру. Например, если добавить параметр **type=slidecomment**, то будут возвращены только объекты, с указанным значением **type**.
- usrsesid – идентификатор сессии пользователя, полученный с помощью авторизации по API (обязательный параметр).

Выходные данные:

- Массив объектов с параметрами модели объекта пользователя. Данные возвращаются постранично.

Изменение информации об объекте пользователя

Тип запроса: PUT

/service/v2/myProfile/objects/{id}

Параметр URL (обязательный):

- id – идентификатор объекта пользователя.

Входные параметры:

- Набор из параметров модели объекта пользователя, которые требуется изменить.
- usrsesid – идентификатор сессии пользователя, полученный с помощью авторизации по API (обязательный параметр).

Модуль Vacancy (Вакансии)

Информация о вакансии представляется следующей моделью данных:

Название поля	Описание	Тип
vacid	Идентификатор вакансии	Число
vactitle	Название вакансии	Строка (255)
vaccode	Код вакансии	Строка (255)
vacownerid	Идентификатор физического лица – Заказчика вакансии	Число
vacowneridname	ФИО физического лица - Заказчика вакансии	Строка (255)
clientid	Идентификатор физического лица - Клиента вакансии	Число
clientidname	ФИО физического лица – Клиента вакансии	Строка (255)
caid	Идентификатор организации вакансии	Число
caidname	Название организации вакансии	Строка (255)
ccid	Идентификатор центра затрат вакансии	Число
ccidname	Название центра затрат вакансии	Строка (255)
rspostid	Идентификатор должности вакансии	Число
rspostidname	Название должности вакансии	Строка (255)
statepostid	Идентификатор штатной должности вакансии	Число
statepostidname	Название штатной должности вакансии	Строка (255)
statusid	Идентификатор статуса вакансии	Малое
statusidname	Название статуса вакансии	Строка (255)

Добавление вакансии

Тип запроса: POST

/service/v2/Vacancy

Входные параметры:

- Набор из параметров модели вакансии. Обязательным параметром является vactitle.

Выходные данные:

- Параметры из модели созданной вакансии.

Получение информации о вакансии

Тип запроса: GET

/service/v2/Vacancy/{id}

Параметр URL (обязательный):

- id – идентификатор вакансии.

Выходные данные:

- Параметры из модели вакансии.

Изменение информации о вакансии

Тип запроса: PUT

/service/v2/Vacancy/{id}

Параметр URL (обязательный):

- id – идентификатор вакансии.

Входные параметры:

- Набор из параметров модели вакансии, которые требуется изменить.

Выходные данные:

- Параметры из модели вакансии.

Удаление вакансии

Тип запроса: DELETE

/service/v2/Vacancy/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой вакансии.

Модуль qua (Компетенции)

Информация о компетенции представляется следующей моделью данных:

Название поля	Описание	Тип
quaid	Идентификатор компетенции	Строка
quacode	Код компетенции	Строка
quaname	Название компетенции	Строка
quadesc	Описание компетенции	Строка
quaactualdays	Срок актуальности, месяцев	Строка
quatype	Тип. Возможные значения: • «0» - «Авто» • «1» – «Пользовательская»	Строка
categoryrsid	Категория компетенции (идентификатор)	Строка
categoryrsidname	Категория компетенции (название)	Строка

Добавление компетенции

Тип запроса: POST

/service/v2/qua

Входные параметры:

- Набор из параметров модели компетенции. Обязательным параметром является quaname.

Выходные данные:

- Параметры из модели созданной компетенции.

Получение информации о компетенции

Тип запроса: GET

/service/v2/qua/{id}

Параметр URL (обязательный):

- id – идентификатор компетенции.

Выходные данные:

- Параметры из модели компетенции.

Изменение информации о компетенции

Тип запроса: PUT

/service/v2/qua/{id}

Параметр URL (обязательный):

- id – идентификатор компетенции.

Входные параметры:

- Набор из параметров модели компетенции, которые требуется изменить.

Выходные данные:

- Параметры из модели компетенции.

Удаление компетенции

Тип запроса: DELETE

/service/v2/qua/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой компетенции.

Получение списка мероприятий компетенции

Тип запроса: GET

/service/v2/quameasures

Метод возвращает список мероприятий, присваивающих компетенцию, аналогично списку «Мероприятия» на экране компетенции.

Входные параметры (обязательные):

- quaId – идентификатор компетенции, список мероприятий которой нужно получить.

Выходные данные:

- Список мероприятий с параметрами из модели мероприятия.

Модуль personAttainments (Присвоенные достижения)

Модуль позволят присваивать физическим лицам достижения (создавать присвоенные достижения).
Информация о присвоенном достижении представляется следующей моделью данных:

Название поля	Описание	Тип
atnmid	Идентификатор достижения (поле Автономер на странице достижения)	Число
personid	Идентификатор физического лица (поле Автономер на странице физического лица)	Число
patnmcomment	Комментарий	Строка(255)
patnmdate	Дата присвоения достижения	Дата

На заметку

– *При создании присвоенного достижения по API в поле Источник присвоения устанавливается значение «Внешняя система».*

Добавление присвоенного достижения

Тип запроса: POST

/service/v2/personAttainments

Входные параметры:

- Набор из параметров модели присвоенного достижения. Обязательным параметром является atnmid (идентификатор достижение, которое присваивается) и personid (идентификатор физического лица, которому присваивается достижение).

На заметку

– *Если параметр patnmdate не передан, то в качестве даты создания используется системная дата и время создания присвоенного достижения.*

– *Если передан токен аутентификации пользователя usrsesid, то в качестве пользователя, создавшего присвоенное достижение, будет сохранен пользователь, которому принадлежит токен.*

Выходные данные:

- Параметры из модели созданного присвоенного достижения.

Модуль gameAccountChanges (Изменения игрового счета)

Модуль позволяет добавлять физическим лицам изменения игрового счета
Информация об изменении игрового счета представляется следующей моделью данных:

Название поля	Описание	Тип
personid	Идентификатор физического лица (поле Автономер на странице физического лица)	Число
comment	Комментарий	Строка (255)

Данные на вкладке Валюты в изменении игрового счета представляются следующей моделью данных:

Название поля	Описание	Тип
autonumber	Идентификатор валюты (поле Автономер на странице значения справочника Игровые валюты)	Число
amount	Сумма	Число

На заметку

- *Данные для записей вкладки Валюты передаются в теле запроса и не учитываются при расчёте подписи.*
- *Данные передаются с заголовком Body content type = application/json*

Например

[{"autonumber":"-42","amount":100},
{"autonumber":"14764","amount":300},
{"autonumber":"-41","amount":200}]

Добавление изменения игрового счета

Тип запроса: POST

/service/v2/gameAccountChanges/{personId}

Параметр URL (обязательный):

- personid – идентификатор физического лица.

Входные параметры:

- Параметры модели изменения игрового счета.
- Json массив записей валют изменения игрового счета

Выходные данные:

- Параметры из модели изменения игрового счета.

На заметку

- *Если для пользователя не создан игровой счет перед добавлением изменения игрового счета по API, то игровой счет создается автоматически.*

Модуль currentGameAccount (Игровой счёт пользователя)

Модуль позволяет получать общие данные игрового счёта для физического лица, сессия которого передана в качестве входного параметра. Указание подписи запроса и идентификатора приложения не требуется. Информация об игровом счете представляется следующей моделью данных:

Название поля	Описание	Тип
userid	Идентификатор физического лица (поле Автономер на странице физического лица)	Число
userFio	ФИО физического лица, по которому возвращен игровой счёт	Строка
balanceBeans	Json-массив вида: [{"rsId":"10","rsName":"Монеты","amount":123}, {"rsId":"20","rsName":"Опыт","amount":234}, {"rsId":"30","rsName":"Баллы","amount":345}] Где: • rsId – идентификатор значения справочника «Игровые валюты»;	Массив полей: rsId - Число rsName – Строка (255) amount - Число

	• rsName – название значения справочника «Игровые валюты»; • amount - количество набранной пользователем игровой валюты.	

Получение игрового счёта пользователя

Тип запроса: GET

/service/v2/currentGameAccount

Входной параметр (обязательный):

- «usrsesid» - сессия пользователя, полученная с помощью метода auth.

Выходные данные:

- Параметры из модели игрового счета.

Модуль GroupVacancy (Группы вакансий)

Информация о группах вакансий представляется следующей моделью данных:

Название поля	Описание	Тип
gcid	Идентификатор группы	Число
gcname	Название группы	Строка(100)
gcdesc	Описание группы	Строка(255)
creatorid	Идентификатор физлица – Создатель группы	Число
creatoridname	ФИО физлица – Создатель группы	Строка(255)
grkind	Вид группы (0 – Динамическая, 1 – Полудинамическая, 2 – Статическая);	Малое

Создание группы вакансий

Тип запроса: POST

/service/v2/GroupVacancy

Входные параметры:

- Параметры из модели группы (обязательное – gcname).

Выходные данные:

- Параметры из модели созданной группы.

На заметку

- *Если не указан вид группы, то по умолчанию создается статическая группа.*
- *Всегда создается общая группа.*

Получение информации о группе вакансий

Тип запроса: GET

/service/v2/GroupVacancy/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Параметры из модели группы.

Удаление группы вакансий

Тип запроса: DELETE

/service/v2/GroupVacancy/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

На заметку

– *Группа вакансий не может быть удалена, если содержит дочерние группы вакансий.*

Изменение группы вакансий

Тип запроса: PUT

/service/v2/GroupVacancy/{id}

Параметр URL (обязательный):

- id – идентификатор группы.

Входные параметры:

- Набор из параметров модели группы вакансий, которые нужно изменить.

Выходные данные:

- Параметры из модели группы вакансий.

Добавление дочерней группы

Тип запроса: POST

/service/v2/GroupVacancy/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор добавляемой группы.

Исключение дочерней группы

Тип запроса: DELETE

/service/v2/GroupVacancy/{id}/childGroups/{childId}

Параметры URL (обязательные):

- id – идентификатор группы.
- childId – идентификатор исключаемой группы.

Получение списка дочерних групп

Тип запроса: GET

/service/v2/GroupVacancy/{id}/childGroups

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели группы. Данные возвращаются постранично.

Добавление вакансии в группу

Тип запроса: POST

/service/v2/GroupVacancy/{id}/childVacancies/{vacId}

Параметры URL (обязательные):

- id – идентификатор группы.
- vacId – идентификатор вакансии.

На заметку

- *Добавлять элементы возможно только в группы вида «2» - статические.*
- *В полудинамические группы элементы добавляются фоновым заданием на основе настроек фильтра в группе, которые задаются через интерфейс системы.*
- *В динамические группы элементы не добавляются. Список элементов для отображения рассчитывается при обращении к группе на основе заданного фильтра.*
- *Настройка полудинамических и динамических групп через API не поддерживается.*

Исключение вакансии из группы

Тип запроса: DELETE

/service/v2/GroupVacancy/{id}/childVacancies/{vacId}

Параметры URL (обязательные):

- id – идентификатор группы.
- vacId – идентификатор вакансии.

Получение списка вакансий, входящих в группу

Тип запроса: GET

/service/v2/GroupVacancy/{id}/childVacancies

Параметр URL (обязательный):

- id – идентификатор группы.

Выходные данные:

- Массив объектов с параметрами модели вакансии. Данные возвращаются постранично.

Модуль externalRecords (Внешние записи)

Модуль позволяет хранить ссылки на записи мероприятий и связывать эти ссылки с мероприятиями в системе.
Информация о внешней записи представляется следующей моделью данных:

Название поля	Описание	Тип
erid	Идентификатор внешней записи	Строка
ername	Название. Обязательное поле	Строка (200)
erdescription	Описание	Строка (1000)
ersize	Размер	Число
erstartdate	Дата начала	Дата со временем
erenddate	Дата окончания	Дата со временем
ercode	Код	Строка (100)
erlink	Ссылка	Строка (1000)
ermeid	Мероприятие. В поле возвращается идентификатор мероприятия	Строка
ermeidname	Название мероприятия, идентификатор которого возвращается в ermeid. Используется только при получении данных о внешней записи	Строка

Получение всех внешних записей

Тип запроса: GET

/service/v2/externalRecords

Входные параметры (необязательные):

- Параметры внешней записи, по которым требуется осуществить фильтрацию (поиск) возвращаемых данных.

Выходные данные:

- Параметры внешней записи.

Добавление внешней записи

Тип запроса: POST

/service/v2/externalRecords

Входные параметры:

- Набор из параметров внешней записи. Обязательным параметром является ername.

Выходные данные:

- Параметры созданной внешней записи.

Получение информации о внешней записи

Тип запроса: GET

/service/v2/externalRecords/{id}

Параметр URL (обязательный):

- id – идентификатор внешней записи.

Выходные данные:

- Параметры внешней записи.

Изменение внешней записи

Тип запроса: PUT

/service/v2/externalRecords/{id}

Параметр URL (обязательный):

- id – идентификатор внешней записи.

Входные параметры:

- Набор параметров внешней записи, которые требуется изменить.

Выходные данные:

- Параметры внешней записи.

Удаление внешней записи

Тип запроса: DELETE

/service/v2/externalRecords/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемой внешней записи.

Модуль ListConstructor (Динамические списки)

Модуль позволяет получать по API данные из динамических списков. API не позволяет создавать, изменять и удалять динамические списки.
Информация о динамических списках представляется следующей моделью данных:

Название поля	Описание	Тип
dlid	Идентификатор динамического списка	Число
dlname	Название динамического списка	Строка (255)

Для каждого динамического списка модель данных, содержащихся в этом списке, различается и зависит от настройки динамического списка.

Получение перечня динамических списков

Тип запроса: GET

/service/v2/ListConstructor

Входные параметры:

- Нет.

Выходные данные:

- Массив объектов с параметрами модели динамических списков. Данные возвращаются постранично.

Получение содержимого динамического списка

Тип запроса: GET

/service/v2/ListConstructor/{dlid}/data

Входные параметры:

- dlid – идентификатор динамического списка.

Выходные данные:

- Массив объектов с параметрами, настроенными в динамическом списке с переданным id. Набор параметров списка и названия параметров списка настраиваются на вкладке Колонки в карточке динамического списка при настройке списка в интерфейсе системы. Данные возвращаются постранично.

На заметку

- *Содержимое динамического списка вычисляется в момент запроса в соответствии с правилами, настроенными для этого динамического списка (Источники данных, Фильтр, Колонки).*
- *Данные, возвращаемые при запросе из динамического списка, представляют собой таблицу. Каждая строка таблицы является JSON-объектом. Поля этого объекта – колонки динамического списка.*
- *Для настройки имен полей необходимо указать в интерфейсе системы требуемые имена в качестве значения полей «Идентификатор для API» для колонок динамического списка (вкладка Колонки). Если значение Идентификатор для API для колонки не задано, то в качестве имени соответствующего параметра в ответе на запрос будет возвращаться техническое внутренне имя колонки (например, col_dd26f585).*
- *Правила, настроенные на вкладке Фильтр применяются всегда. С помощью API нельзя включать/отключать эти правила при запросе. Запрос может содержать правила для дополнительной фильтрации содержимого динамического списка: правила поиска объектов и расширенные правила фильтрации. Таким образом сперва применяются правила фильтрации, настроенные на вкладке Фильтр, после этого правила поиска и после этого расширенные правила фильтрации.*
- *Для фильтрации содержимого динамического списка возможно использовать имена параметров, заданных как значения Идентификатор для API для колонок списка.*

Модуль events (События)

Информация о событии представляется следующей моделью данных:

Название поля	Описание	Тип
evid	Идентификатор события	Число
etid	Идентификатор типа события. Обязательное поле	Число
etidname	Тип события	Строка (255)
evstatusid	Идентификатор статуса события	Число
evstatusidname	Статус события	Строка (255)
creatorid	Идентификатор создателя события	Число
creatoridname	Создатель события (ФИО физ. лица)	Строка (500)
evrespid	Идентификатор ответственного по событию	Число
evrespidname	Ответственный за событие (ФИО физ. лица)	Строка (500)
canid	Идентификатор кандидата	Число

canidname	Кандидат (ФИО кандидата)	Строка (500)
vacid	Идентификатор вакансии события	Число
vacidname	Вакансия события (Название)	Строка (255)
createdate	Дата создания	Дата
updatedate	Дата обновления	Дата

Добавление события

Тип запроса: POST

/service/v2/events

Входные параметры:

- Набор из параметров модели события. Обязательным параметром является etid.

Выходные данные:

- Параметры из модели созданного события.

Получение информации о событии

Тип запроса: GET

/service/v2/events/{id}

Параметр URL (обязательный):

- id – идентификатор события.

Выходные данные:

- Параметры из модели события.

Изменение информации о событии

Тип запроса: PUT

/service/v2/events/{id}

Параметр URL (обязательный):

- id – идентификатор события.

Входные параметры:

- Набор из параметров модели события, которые требуется изменить.

Выходные данные:

- Параметры из модели события.

Удаление события

Тип запроса: DELETE

/service/v2/events/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого события.

Модуль candidates (Кандидаты)

Информация о кандидате представляется следующей моделью данных:

Название поля	Описание	Тип
canid	Идентификатор кандидата	Число
canfio	ФИО кандидата	Строка (500)
canstatusid	Статус кандидата (id)	Число
canstatusidname	Статус кандидата (название)	Строка (255)
cantype	Тип кандидата («0» - Внешний кандидат, «1» - Внутренний кандидат	Малое
cansex	Пол «0» - Мужской, «1» - Женский, «2» - Неважно	Малое
ccemail	Email кандидата	Строка (255)

Добавление кандидата

Тип запроса: POST

/service/v2/candidates

Входные параметры:

- Набор из параметров модели кандидата. Обязательным является поле «Имя» - «canfirstname».

Выходные данные:

- Параметры из модели созданного кандидата.

Получение информации о кандидате

Тип запроса: GET

/service/v2/candidates/{id}

Параметр URL (обязательный):

- id – идентификатор кандидата.

Выходные данные:

- Параметры из модели кандидата.

Изменение информации о кандидате

Тип запроса: PUT

/service/v2/candidates/{id}

Параметр URL (обязательный):

- id – идентификатор кандидата.

Входные параметры:

- Набор из параметров модели кандидата, которые требуется изменить.

Выходные данные:

- Параметры из модели кандидата.

Удаление кандидата

Тип запроса: DELETE

/service/v2/candidates/{id}

Параметр URL (обязательный):

- id – идентификатор удаляемого кандидата.

Модуль rubricator (Справочник)

Информация о значении справочника представляется следующей моделью данных:

Название поля	Описание	Тип
rsid	Id значения справочника	Число
rscode	Код значения справочника	Строка (100)
rsname	Название значения справочника	Строка (500)

Получение списка активных значений справочников

Метод позволяет получить активные значения справочника по id рубрики справочника для возможности выбора значений на внешнем сайте аналогично лукап-полям в системе Мираполис. Активными являются значения, в настройках которых отмечен признак «Активно». В ответе возвращаются все активные значения справочника, без постраничной выдачи.

Тип запроса: GET

/service/v2/rubricator/{rubId}/rs

Параметр URL (обязательный):

- rubid – идентификатор рубрики справочника, активные значения которой нужно получить.

Выходные данные:

- Параметры из модели значения справочника.

Модуль educationDirections (Направления обучения)

Модуль позволяет получить значения справочника Направления обучения, а также количество объектов по каждому направлению обучения, которые доступны пользователю, авторизация которого передается в запрос.

Информация о значении справочника направлений обучения представляется следующей моделью данных:

Название поля	Описание	Тип
id	Идентификатор направления обучения	Строка
name	Название направления обучения	Строка (500)
allowchoosevalue	Разрешен ли выбор значения (“false» – нет, «true» –да)	Логическое
parenteducationdirections	Список идентификаторов «Активных» родительских направлений обучения. Массив полей «id» (строка), содержащих идентификатор «id» родительского направления обучения.	Массив

childeducationdirections	Список идентификаторов «Активных» дочерних направлений обучения. Массив полей «id» (строка), содержащих идентификатор «id» родительского направления обучения.	Массив
objectscount	Количество доступных пользователю объектов по направлению обучения	Число
code	Код	Строка (100)
Isdefault	Является значением по умолчанию	Логическое
backgroundcolor	Цвет фона. Поле возвращается, если заполнено в значении справочника. Код цвета в шестнадцатеричной форме.	Строка (6)

Получение списка активных значений направлений обучения

Тип запроса: GET

/service/v2/educationDirections

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.

Выходные данные:

- Список активных направлений обучения (в настройках которого включено значение “Активно”), доступных пользователю, авторизация которого передана по API. Возвращаются все доступные значения, без постраничной выдачи

Модуль knowledgeBaseList (База знаний)

Модуль позволяет получить записи базы знаний, доступные пользователю, чья авторизация передана в запрос.
Информация о записи базы знаний представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
mehoursminutes	Значение поля «Объем мероприятия (часы)». Значение в секундах	Дробное число
metheoryduration	Значение поля «Теория (часов)». Значение в секундах	Дробное число
mepracticeduration	Значение поля «Практика (часов)». Значение в секундах	Дробное число
bookmarked	Добавленно текущим пользователем в избранное. Имеет значение «true», если объект внесен в избранное текущего пользователя, «false» - если нет.	Логическое
assigned	Назначенное пользователю («false» – нет, «true» -да). Имеет значение true, если текущий пользователь является участником данного мероприятия.	Логическое
passed	Пройдено («false» – нет, «true» -да). Имеет значение «true», если текущий пользователь является участником мероприятия и имеет статус прохождения “Пройдено”.	Логическое
Records	Массив записей вебинара (описание массива в таблице после текущей)	Массив
id	Идентификатор объекта, тип которого вернулся в поля type	Строка
type	Системное название типа объекта: • «studentcourse» – мероприятие (в том числе типовое) • «mediapreview» – ресурс • «ViewNews» – публикация	Строка
contenttype	Вид объекта: • «0» – Вебинары • «1» – Видео • «2» – Курсы и тесты • «3» – Материалы для чтения	Строка
typetitle	Название типа объекта. Для мероприятий могут быть значения:	Строка

	• «Электронный курс» • «Программа мероприятий» • «Практическое задание» • «Электронный тест» • «Вебинар» • «Очное мероприятие» • «Заочное мероприятие» • «Расписание» Для ресурсов могут быть значения: • «Файл» • «Wiki» • «Ссылка на внешний ресурс» • «Встраиваемый внешний ресурс» Для публикаций могут быть значения: • «Новость» • «Объявление»	
name	Название объекта	Строка
cover	Ссылка для скачивания обложи направления обучения. Для скачивания обложки необходимо добавить к возвращенной ссылке авторизацию пользователя, полученную по API	Строка
tagset	Метки. Массив записей о метках вида: • "tagset": "id" – идентификатор метки • "tagsetname": "idname" – название метки	Массив
directionset	Направления обучения. Массив записей о направлениях обучения вида: • "directionset": "id" – идентификатор направления обучения • "directionsetname": "idname" – название направления обучения	Массив
mestartdate	Дата начала мероприятия	Дата со временем
meenddate	Дата окончания мероприятия	Дата со временем
pubdate	Дата публикации (для ресурсов и публикаций)	Дата
pubtype	Тип публикации. «0» - Объявление. «1» - Новость.	Строка
rkind	Вид ресурса. «0» - Файл. «1» - Ссылка.	Малое
cost	Стоимость мероприятия	Строка
mecontenttype	Тип мероприятия: • «0» – Электронный курс • «1» – Программа мероприятий • «3» – Практическое задание • «4» – Электронный тест • «5» – Вебинар • «6» – Очное мероприятие • «7» – Заочное мероприятие • «8» – Расписание • «9» – Техническая учеба	Малое
resourcefilebytesize	Размер файла в байтах	Число
iscomment	Разрешено комментирование объекта («false» – нет, «true» -да). Имеет значение true, если текущий объект разрешено комментировать.	Логическое
views	Количество просмотров объекта	Число

Информация о записи вебинара представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
id	Идентификатор записи	Строка
name	Название мероприятия во время записи	Строка
period	Период ведения записи в формате [Дата и время начала записи] - [Дата и время завершения записи]	Строка
duration	Длительность записи в формате hh:mm:ss	Строка

viewLink	Ссылка для просмотра записи	Строка
downloadLink	Ссылка на скачивание записи в формате mvror	Строка
downloadVideoLink	Ссылка на скачивание mp4	Строка

Получение списка объектов базы знаний текущего пользователя с дополнительными параметрами

Тип запроса: GET

/service/v2/knowledgeBaseList

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Список направлений обучения (опционально). В параметре **directionIds** возможно указать список id направлений обучения через запятую. В результате запроса будут только записи, которые относятся к переданным направлениям обучения.
- Список видов контента(опционально). В параметре **contentTypeIds** возможно указать список id видов записей через запятую. В результате запроса будут только записи, которые относятся к переданным видам контента.
- Список меток (опционально). В параметре **tagIds** возможно указать список id меток через запятую. В результате запроса будут только записи, которые относятся к переданным меткам.
- Направление сортировки (опционально) – sort (по убыванию – desc, по возрастанию – asc). Сортировка задается в виде «sort=contenttype:asc». Поля, доступные для сортировки:
    - По полю «Индекс лояльности» - «index_event».
    - По полю «Дата публикации» - «pubdate» (для ресурсов и публикаций).
    - По полю «Название» – «name».
    - По полю «Вид» – contenttype.
    - По полю «Количество прохождений» – passessort.

Выходные данные:

- Список записей базы знаний, выдача результатов постраничная

Модуль mediausergr (Доступные пользовательские группы ресурсов)

Модуль позволяет получить список пользовательских групп ресурсов, доступных пользователю, чья авторизация передана в запрос.
Информация о пользовательских группах ресурсов авторизованного пользователя представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
ugrid	Идентификатор пользовательской группы	Число
ugrname	Название пользовательской группы	Строка (255)
ugrdesc	Стандартное описание пользовательской группы	Строка (4000)
coverfilepath	Ссылка на скачивание обложки. Для скачивания файла обложки, в заголовок запроса access_token необходимо добавлять авторизацию пользователя по API	Строка

Дополнительные поля, доступные для запроса при запросе в bean_add_fields:

Название поля	**Описание**	**Тип**
ugrparentid	Идентификатор родительской пользовательской группы.	Число
ugrextdesc	Расширенное описание пользовательской группы	Строка (1048576)
personid	Идентификатор создателя пользовательской группы	Число

personidname	ФИО создателя пользовательской группы	Строка
ugrcreatedate	Дата создания пользовательской группы	Дата
ugrshowdesc	Показывать описания группы («0» – Не показывать, «1» – Показывать стандартное, «2» – Показывать расширенное)	Малое число
ugrshowsubgroup	Показывать подгруппы («0» – Не показывать, «1» – Первого уровня, «2» – Всех подуровней, «3» - заданное число, «4» – Показывать подгруппы как списки с категориями, «5» – Показать, как список, «6» – Показать подгруппы как списки без категорий)	Малое число
ugrshowsubgroupimage	Показывать изображения подгрупп	Логическое
ugrshowsubgroupdesc	Показывать описания подгрупп («0» – Не показывать, «1» – Показывать стандартные, «2» – Показывать расширенные)	Малое число
ugrsubgrouplevelcount	Количество показываемых подуровней	Число
ugrorder	Порядок	Число
ugrcolumn	Количество колонок	Число
ugrsortorder	Сортировать элементы («0» – По алфавиту, «1» – По дате добавления)	Малое число
ugrsortdirection	Порядок сортировки («0» – По возрастанию, «1» – По убыванию)	Малое число

Получение пользовательских групп ресурсов пользователя

Тип запроса: GET

/service/v2/mediausergr

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.

Выходные данные:

- Список объектов, доступных текущему пользователю, с учетом входного фильтра и сортировки. Выдача результатов – постраничная.

Модуль resources (Доступные ресурсы)

Модуль позволяет получить список ресурсов, доступных пользователю, чья авторизация передана в запрос. Информация о ресурсе представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
rid	Идентификатор ресурса	Строка
rname	Название ресурса	Строка(255)
rdesc	Описание ресурса	Строка(4000)
rcreatedate	Дата создания ресурса	Дата
coverfilepath	Ссылка на скачивание обложки. Для скачивания файла обложки, в заголовок запроса access_token необходимо добавлять авторизацию пользователя по API	Строка
passes	Количество просмотров ресурса	Строка

Дополнительные поля, доступные для запроса при запросе в bean_add_fields:

Название поля	**Описание**	**Тип**
rchangedate	Дата изменения ресурса	Дата
rkeywords	Ключевые слова	Строка(255)
fileid	Идентификатор файла (только для ресурса вида Файл). Для скачивания файла необходимо сформировать ссылку вида: *https://URL/mira/Do?doaction=DownloadFileAction&fileId=<fileid>*, заменив <fileid> на присланное значение fileid и добавив параметр usrsesid с идентификатором сессии, полученном через модуль auth.	Файл

rurl	Ссылка на внешний ресурс (только для ресурса вида Внешний ресурс)	Строка(1000)
rexternaldata	Код для вставки (только для ресурса вида Встраиваемый внешний ресурс)	Строка
personid	Идентификатор пользователя, создавшего ресурс	Строка
personidname	ФИО пользователя, создавшего ресурс	Строка
rkind	Вид ресурса («0» - Файл, «1» - Wiki, «2» - Ссылка на внешний ресурс, «3» - Встраиваемый внешний ресурс)	Малое число
risdownload	Доступ к файлу («0» - не доступен, «1» - доступен)	Малое число
statusrsid	Идентификатор статуса ресурса	Строка
statusrsidname	Название статуса ресурса	Строка (100)
checkpersonid	Идентификатор пользователя, изменившего статус	Строка
checkpersonidname	ФИО пользователя, изменившего статус	Строка
rpreview	Предпросмотр («0» - Авто, «1» – Из файлов предпросмотра, «2» - Выключен)	Строка
riscomment	Разрешить оценивание и комментарии	Строка
contentid	Идентификатор контента	Строка
contentidname	Название контента	Строка
rversionaccessedit	Управление версиями («0» - Все, «1» - Модераторы, «2» - Авторы и модераторы)	Строка
rversionaccessview	Доступ к просмотру версий («0» - Все, «1» - Модераторы, «2» - Авторы и модераторы)	Строка
rrelevancedate	Дата актуальности	Дата
rstatisticenabled	Включена ли статистика («0» – Нет, «1» – Да)	Малое
rchangeauthor	Идентификатор пользователя, изменившего текущую версию	Строка
rchangeauthorname	ФИО пользователя, изменившего текущую версию	Строка
donotconvert	Не конвертировать	Логическое
rallowprinting	Разрешить печать	Логическое

Получение списка ресурсов пользователя

Тип запроса: GET

/service/v2/rubricator/{rubId}/rs

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Направление сортировки (опционально) – sort (по убыванию – desc, по возрастанию – asc). Сортировка задается в виде «sort=rname:asc».

Выходные данные:

- Список объектов, доступных текущему пользователю, с учетом входного фильтра и сортировки. Выдача результатов – постраничная.

Модуль publications (Доступные публикации)

Модуль позволяет получить список публикаций, доступных пользователю, чья авторизация передана в запрос.
Информация о публикации представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
pubid	Идентификатор публикации	Строка
pubtype	Тип публикации («0» - Объявление, «1» - Новость)	Строка
pubname	Название публикации	Строка(255)
pubdate	Дата размещения публикации	Дата

pubdata	Текст публикации	Строка
coverfilepath	Ссылка на скачивание обложки. Для скачивания файла обложки, в заголовок запроса access_token необходимо добавлять авторизацию пользователя по API	Строка
passes	Количество прохождений (количество просмотров публикации)	Число

Дополнительные поля, доступные для запроса при запросе в bean_add_fields:

Название поля	**Описание**	**Тип**
pubstatus	Статус публикации («0» – Действительная, «1» - Закрыта)	Малое число
actdays	Срок актуальности публикации в днях	Число
pubiscommentmark	Разрешить комментарии и оценивание	Логическое
pubviewcount	Количество просмотров	Число
keywords	Метки для фильтра	Строка
searchkeywords	Метки для поиска	Строка
showsearchkeywords	Показывать метки для поиска	Логическое
pubdirections	Направления обучения	Массив
pubannounce	Анонс публикации	Строка
pubtextannouncement	Текстовый анонс	Строка
personid	Идентификатор пользователя, создавшего публикацию	Число
personidname	ФИО пользователя, создавшего публикацию	Строка
pubcreatedate	Дата создания публикации	Дата

Получение списка публикаций пользователя

Тип запроса: GET

/service/v2/publications

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Направление сортировки (опционально) – sort (по убыванию – desc, по возрастанию – asc). Сортировка задается в виде «sort=pubname:asc».

Выходные данные:

- Список объектов, доступных текущему пользователю, с учетом входного фильтра и сортировки. Выдача результатов – постраничная.

Модуль usergm (Доступные пользовательские группы мероприятий)

Модуль позволяет получить список пользовательских групп мероприятий, доступных пользователю, чья авторизация передана в запрос.
Информация о пользовательских группах мероприятий представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
ugrid	Идентификатор пользовательской группы	Число
ugrname	Название пользовательской группы	Строка
ugrdesc	Стандартное описание пользовательской группы	Строка
coverfilepath	Ссылка на скачивание обложки. Для скачивания файла обложки, в заголовок запроса access_token необходимо добавлять авторизацию пользователя по API	Строка

Дополнительные поля, доступные для запроса при запросе в bean_add_fields:

Название поля	**Описание**	**Тип**

ugrparentid	Идентификатор родительской пользовательской группы. Обязательное поле. Если нужно разместить группу «в корне» пользовательских групп, нужно указать значение «-10», соответствующее системной пользовательской группе «ПКР».	Число
ugrextdesc	Расширенное описание пользовательской группы	Строка
personid	Идентификатор создателя пользовательской группы	Число
personidname	ФИО создателя пользовательской группы	Строка
ugrcreatedate	Дата создания пользовательской группы	Дата
ugrshowdesc	Показывать описания группы («0» – Не показывать, «1» – Показывать стандартное, «2» – Показывать расширенное)	Малое число
ugrshowsubgroup	Показывать подгруппы («0» – Не показывать, «1» – Первого уровня, «2» – Всех подуровней, «3» - заданное число, «4» – Показывать подгруппы как списки с категориями, «5» – Показать, как список, «6» – Показать подгруппы как списки без категорий)	Малое число
ugrshowsubgroupimage	Показывать изображения подгрупп	Логическое
ugrshowsubgroupdesc	Показывать описания подгрупп («0» – Не показывать, «1» – Показывать стандартные, «2» – Показывать расширенные)	Малое число
ugrsubgrouplevelcount	Количество показываемых подуровней	Число
ugrorder	Порядок	Число
ugrcolumn	Количество колонок	Число
ugrsortorder	Сортировать элементы («0» – По алфавиту, «1» – По дате добавления)	Малое число
ugrsortdirection	Порядок сортировки («0» – По возрастанию, «1» – По убыванию)	Малое число

Получение списка пользовательских групп мероприятий пользователя

Тип запроса: GET

/service/v2/usergm

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.
- Направление сортировки (опционально) – sort (по убыванию – desc, по возрастанию – asc). Сортировка задается в виде «sort=ugrname:asc».

Выходные данные:

- Список объектов, доступных текущему пользователю, с учетом входного фильтра и сортировки. Выдача результатов – постраничная.

Модуль availableMeasures (Доступные мероприятия)

Модуль позволяет получить список мероприятий, доступных пользователю, чья авторизация передана в запрос.
Информация о мероприятии представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
meid	Идентификатор мероприятия	Строка
mename	Название	Строка
medescription	Описание мероприятия	Строка
metype	Тип мероприятия: • «0» – «Типовое мероприятие» • «1» - «Мероприятие» • «2» «Внутренне мероприятие»	Строка
mecode	Код мероприятия	Строка
mestatus	Статус мероприятия: • «1» – «Запланировано»	Строка

	• «2» – «Действительно» • «3» - «Отменено» • «4»- «В архиве»	
mestartdate	Дата и время начала мероприятия	Дата
meenddate	Дата и время окончания мероприятия	Дата
meeduform	Форма проведения: • «0» – «Очная» • «1» – «Дистанционная» • «2» - «Смешанная» • «3» – «Заочная»	Строка
mecontenttype	Вид мероприятия: • «0» – «Электронный курс» • «1» – «Программа мероприятий» • «3» – «Практическое задание» • «4» – «Электронный тест» • «5» – «Вебинар» • «6» – «Очное мероприятие» • «7» – «Заочное мероприятие»	Строка
testid	Идентификатор электронного теста (только для мероприятий вида Электронный тест).	Строка
testidname	Название электронного теста (только для мероприятий вида Электронный тест).	Строка
coverfilepath	Ссылка на скачивание обложки. Для скачивания файла обложки, в заголовок запроса access_token необходимо добавлять авторизацию пользователя по API	Строка
mehoursminutes	Объем мероприятия (часы). Значение в секундах	
metheoryduration	Теория (часов). Значение в секундах	
mepracticeduration	Практика (часов). Значение в секундах	
mepasses	Количество прохождений мероприятия. Для мероприятий всех типов кроме Вебинара – это количество участников мероприятия, которые имею статус прохождения "Пройдено". Для мероприятий типа Вебинар – это количество участников мероприятия, в результатах которых значение поля "Затраченное время" больше 0. Для типовых мероприятий берется сумма значений количества прохождений по всем дочерним мероприятиям.	Число
records	Список записей вебинара	Массив

Информация о записи вебинара представляется следующей моделью данных:

Название поля	**Описание**	**Тип**
id	Идентификатор записи	Строка
name	Название записи	Строка
period	Период ведения записи в формате [Дата и время начала записи] - [Дата и время завершения записи]	Строка
duration	Длительность записи в формате hh:mm:ss	Строка
viewLink	Ссылка для просмотра записи	Строка
downloadLink	Ссылка на скачивание записи в формате mvror	Строка
downloadVideoLink	Ссылка на скачивание mp4	Строка

Получение списка доступных мероприятий

Тип запроса: GET

/service/v2/availableMeasures

Входные параметры:

- В запрос передаётся авторизация пользователя, полученная по API.

- Направление сортировки (опционально) – sort (по убыванию – desc, по возрастанию – asc). Сортировка задается в виде «sort=ugrname:asc».

Выходные данные:

- Список объектов, доступных текущему пользователю, с учетом входного фильтра и сортировки. Выдача результатов – постраничная.

Модуль measureRegistration (Регистрация на мероприятие)

Саморегистрация авторизованного пользователя на мероприятие

Метод позволяет зарегистрировать на мероприятие пользователя, авторизация которого передана в запросе.

Тип запроса: POST

/service/v2/measureRegistration/{measureId}/selfregister

Параметр URL (обязательный):

- measureId – идентификатор мероприятия, на которое нужно зарегистрировать пользователя.

Входные параметры:

- Авторизация пользователя, полученная по API (access_token), обязательный параметр.

Выходные данные:

- Идентификатор созданного участника мероприятия (mmid).

Приложение А. Примеры использования API на PHP

Функция для генерации подписи

```
function signit($url, $params) {// url REST-запроса и массив параметров
    $appid = 'APPID'; //идентификатор приложения
    $secretkey = 'SECRETKEY'; //ключ системы

    $ret_params = $params;//массив передаваемых параметров
    ksort($ret_params); //сортировка параметров по названию
    $ret_params['appid'] = $appid; //помещение в конец массива параметра appid

    $signstring="$url?";//формирование строки для подписи начиная с url
    foreach ($ret_params as $key => $val) {
        if (($val!="")||(gettype($val)!="string")) {
            $signstring .= "$key=$val&";//добавление в строку для подписи очередного параметра
        }
    }
    $signstring .= "secretkey=$secretkey";//дополнение строки для подписи параметром secretkey
    $ret_params['sign'] = strtoupper(md5($signstring)); //формирование ключа и добавление его в
                                                        // массив параметров
    return $ret_params;
}
```

Функция для отправки запроса

```
function sendrequest($url, $parameters, $method, $ret_crange) {
    //дополнение массива параметров значениями appid и sign (используется выше описанная функция signit)
    $curl_data = signit($url, $parameters);

    $ch = curl_init();//инициализация дескриптора запроса
    curl_setopt($ch, CURLOPT_ENCODING, 'UTF-8'); //задание кодировки запроса
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1); //возврат результата
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1); //делает возможным переход на страницу ошибки
    curl_setopt($ch, CURLOPT_HEADER, $ret_crange); //делает возможным возвращение заголовка Content-
Range
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, $method); //задание метода запроса

    $query = http_build_query($curl_data); //построение строки параметров
    switch ($method) {
          case "PUT"://для PUT необходимо передавать длину строки параметров
                    curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Length: " . strlen($query)));
          case "POST"://параметры PUT и POST передаются в теле запроса
                    curl_setopt($ch, CURLOPT_POSTFIELDS, $query);
                    break;
          case "GET"://для GET и DELETE параметры указываются в заголовке
          case "DELETE":
                    $url .= "?$query";
    }
    curl_setopt($ch, CURLOPT_URL, $url); //задание url запроса

    $curl_response = curl_exec($ch); //выполнение запроса
    $response = json_decode($curl_response, true); //парсинг результатов
    if (!$response) $response = $curl_response; //если результат не json
    $code = curl_getinfo($ch, CURLINFO_HTTP_CODE); //получение кода результата
    curl_close($ch);

    //анализ ответа
    if ($code != 200) {
        throw new Exception("Неправильный HTTP-код: ".$code);
    } else if (is_array($response)&&isset($response["errorMessage"]))     {
        throw new Exception("Возвращена ошибка: ".$response["errorMessage"]);
    } else {
        return $response;
    }
}
```

Создание мероприятия

```
$service_url = "https://systemurl.ru/mira/service/v2/measures";

//задание массива параметров мероприятия
$parameters = array(
    'mename' => 'Тестовое мероприятие',
    'mestartdate' => '2013-04-04 13:00:00.000',
    'meenddate' => '2013-04-04 14:00:00.000',
    'metype' => '1',
    'meeduform' => '0'
);

$res = sendrequest($service url, $parameters, "POST", 0);
echo "Создано мероприятие с id $res";
```

Редактирование мероприятия

```
$measureId = …;//id ранее созданного мероприятия
$service url = "https://systemurl.ru/mira/service/v2/measures/$measureId";

//задание массива параметров мероприятия
$parameters = array(
    'mename' => 'Измененное мероприятие',
    'mestartdate' => '2013-04-06 13:00:00.000',
    'meenddate' => '2013-04-06 14:00:00.000',
);

$res = sendrequest($service url, $parameters, "PUT", 0);
echo "Обновлено мероприятие с id $res";
```

Получение информации о мероприятии

```
$measureId = …;//id ранее созданного мероприятия

$service_url = "https://systemurl.ru/mira/service/v2/measures/$measureId";

$res = sendrequest($service_url, array(),"GET", 0);
echo "Параметры мероприятия с id $measureId: ";
print_r $res;
```

Удаление мероприятия

```
$measureId = …;//id ранее созданного мероприятия

$service url = "https://systemurl.ru/mira/service/v2/measures/delete?$measureId";

$res = sendrequest($service_url, array(),"DELETE", 0);
echo "Удалено мероприятие с id $measureId";
```

Получение количества мероприятий из заголовка Content-Range

```
$service_url = "https://systemurl.ru/mira/service/v2/measures";

//отделяем заголовок от тела запроса
$parsed = explode("\r\n\r\n", $res);
$header = $parsed[0];
$body = $parsed[1];
//разбиваем заголовок на строки
$headers = array();
foreach (explode("\n", $header) as $i => $h) {
          $h = explode(':', $h, 2);
          if (isset($h[1])) {
                    $headers[$h[0]] = trim($h[1]);
          }
}

//разбиваем заголовок Content-Range на отдельные элементы
$contentrange=preg_split('/[\s\-\/]/',$headers['Content-Range']);
```

```
//в третьем значении находится количество элементов
$count=$contentrange[3];
print_r($count);
```

Конвертация строк в другую кодировку

При использовании текста с кодовой страницей, отличной от UTF-8 необходимо перед передачей параметров в сервис конвертировать их из исходной кодировки в UTF-8, например, с помощью функции iconv:

```
$text= iconv("UTF-8","cp1251",'Исходный текст');
```

Конвертация параметров запроса GET в подходящий для url формат

При использовании текста в параметрах типа GET его необходимо подготовить для передачи с помощью функции urlencode:

```
$text= urlencode('Исходный текст');
```

История изменений документа

Автор изменений	Описание изменений	Дата изменений	Версия документа	Версия системы
Авторов А.А.	Первая версия документа в новой редакции	20.02.2013	1.00	
Авторов А.А.	Вторая версия документа в новой редакции	22.03.2013	2.00	
Авторов А.А.	Внесена обязательность параметра appid, убрана необходимость параметра secretkey в запросе, оформлен раздел Синтаксис вызова	30.03.2013	2.01	
Авторов А.А.	Дополнительное уточнение, что secretkey не используется в запросе, а только для формирования подписи.	01.04.2013	2.02	
Авторов А.А.	Добавлены примеры на PHP и формат параметров.	04.04.2013	2.03	
Авторов А.А.	Добавлен раздел Настройка системы для работы с API.	09.04.2013	2.04	
Авторов А.А.	Добавлен метод webinarRecords, замечание о пустых значениях и исправление medesc->medescription	18.04.2013	2.05	
Авторов А.А.	Добавлено замечание с описанием возвращаемой ошибки при неправильно указанном адресе системы.	03.05.2013	2.06	
Авторов А.А.	Убрана обязательность передачи идентификатора объекта для запросов на изменение.	01.06.2013	2.07	
Авторов А.А.	Исправлены опечатки, добавлены подсказки по параметрам некоторых методов, исключен метод отправки приглашения преподавателю, добавлен метод регистрации участников по email.	22.06.2013	2.08	
Авторов А.А.	Убран статус мероприятия 0, переименован статус 1(Планируется -> В разработке).	23.07.2013	2.09	
Головенков П.В.	Добавлена новая функция в модуль CA. Добавлен новый модуль GroupMeasure.	30.08.2013	2.10	
Головенков П.В.	Добавлена функция поиска id по параметрам в модуль Access. Исправлено описание функции поиска id по параметрам в модуле CA. Добавлено замечание в формат параметров о формировании подписи. Исправлена ошибка в описании модели мероприятия.	13.09.2013	2.11	
Головенков П.В.	Добавление описания функции search/ids в оставшиеся модули. Добавление параметра ppsex в модуль Person.	01.10.2013	2.12	
Буряков Р.Б.	Добавлена функция получения результатов мероприятия Measure/{measureid}/results.	17.10.2013	2.13	
Авторов А.А.	Добавлена длина и обязательность полей	17.03.2014	2.14	
Авторов А.А.	Дополнены примеры	20.03.2014	2.15	

Головенков П.В.	Добавлена информация о статусе пользователя и видах мероприятий.	15.04.2014	2.16	
Буряков Р.Б.	Исправлено несколько опечаток, добавлено описание метода получения данных о группе мероприятий.	27.05.2014	2.17	
Головенков П.В.	Добавлено описание модуля auth.	06.06.2014	2.18	
Спицын А.Ю.	Документ переработан в связи с введением версионности API и изменением ответов на запросы.	15.07.2014	2.19	
Буряков Р.Б.	Добавлено поле для скачивания записи в видео без создания новой версии инструкции.	25.08.2014	2.19	
Буряков Р.Б.	Добавлен метод Получение списка мероприятий авторизованного пользователя. В описании изменений добавлен столбец Версия системы, отражающий версию, для которой актуальны изменения. В примерах контекст mkc заменен на mira.	05.11.2014	2.20	4.6.11 2014-12-22
Буряков Р.Б.	Введена версия API v2. Изменены названия модулей. Добавлены модули PersonGroup, CaGroup. В модели данных групп указан параметр gcparentidname. Убрано описание, что для удаления группы объектов, не должно быть дочерних объектов. Добавлены модули Report и ReportTemplate.	13.11.2014	2.21	4.6.12 2014-11-13
Буряков Р.Б.	Добавлена подсказка, где отображаются системные названия параметров отчета для модуля ReportTemplate. Добавлен модуль Certificate. Добавлена подсказка, как настроить ссылку на страницу с секретным ключом для сервиса LMS. Добавлено описание, как получать и использовать полный список полей объектов. Добавлены методы получения списка сессий входа в систему и последней сессии входа в систему.	26.11.2014	2.22	4.6.12 2014-11-26
Буряков Р. Б.	Изменены названия модулей в версии API v2 и добавлено название модуля в системе. Модуль Access переименован в модуль roles. Добавлены модули mediaResources, mediaUserGroups, mediaGroups, favorites, certTypes, tests, measureProgress, measureAttempts, measureInteractions, messages, measureUserGroups, comments и galleries. В модуль certificates добавлен метод получения списка сертификатов пользователя. В модуль myMeasures в модель получения мероприятий	13.05.2015	2.23	4.6.12 2015-04-27

	авторизованного пользователя добавлены поля sourceid (id источника) и mmdregistrationtype (тип регистрации), в метод получения списка мероприятий пользователя в качестве участника в возвращаемые данные добавлены contentid, testid и testcsid. В модуль measures в модель мероприятия добавлено поле идентификатора пользовательской группы ресурсов, в методе добавления участника мероприятия добавлен способ задавать тип регистрации, добавлен метод получения ресурсов мероприятия и метод завершения мероприятия. В модуле roles изменен адрес вызова метода v2/roles/byPersonId/{personId} на v2/roles/byPerson/{personId}, добавлены методы для работы с настройкой списка организаций, за которые ответственны пользователи с определенными ролями. В модуле mediaUserGroups в модель пользовательской группы добавлено поле objcount (количество объектов в группе). Методы поиска объектов перенесены в раздел **Стандартные методы** из описания всех модулей. Для метода постраничного получения объектов добавлена возможность фильтрации (поиска). Обновлен пример использования API. Добавлено описание загрузки файлов по API. Исправлено несколько опечаток, описание методов приведено к одному виду с учетом терминологии системы.			
Буряков Р. Б.	Исправлено описание методов **Изменение дат обучения слушателя на мероприятии** и **Изменение дат обучения слушателя на мероприятии, зарегистрированного по источнику**.	14.07.2015	2.24	4.6.12 2015-04-27
Буряков Р. Б.	Добавлен модуль **surveys (Вебинар-опросы)**. В модуль measures добавлен метод **Получение списка вебинар-опросов мероприятия**. Добавлен пример **Получение количества мероприятий из заголовка Content-Range**. Исправлено несколько опечаток.	25.08.2015	2.25	4.6.12 2015-07-31
Буряков Р. Б.	Добавлен модуль **myProfile**. Для модуля **auth** добавлено пояснение, что запросы не требуют подписи sign, а также могут быть вызваны с сервера доставки.	01.09.2015	2.26	4.6.14 2015-09-01
Буряков Р. Б.	Исправлена опечатка в адресе метода **Получение информации об объектах пользователя**. Добавлено предупреждение о том, что в методе mirads/service/auth/login	08.09.2015	2.27	4.6.14 2015-09-01

	на сервере доставки не используется версия API.			
Буряков Р.Б.	Исправлено несколько опечаток. Добавлено предупреждение в метод Открытие пользовательской сессии о том, что получение нового id сессии для пользователя аннулирует ранее полученный id сессии для того же пользователя.	07.12.2016	2.28	4.6.14 2015-09-01
Буряков Р.Б.	Добавлено пояснение о передаваемых полях в метод Добавление ответа на вопрос.	16.08.2017	2.29	4.6.17 2017-08-16
Буряков Р.Б.	Добавлены возможности: расширенной фильтрации при получении списков объектов, работы с вакансиями, группами вакансий, а также получения перечня и данных динамических списков.	16.08.2017	2.29	4.6.17 2017-08-16
Буряков Р.Б.	Актуализация примера на php, актуализация названий параметров, добавление пояснения о формировании подписи при работе системы по нескольким адресам, добавление новых статусов физ. лиц, добавлено пояснение, что не нужно перегенерировать идентификатор сессии пользователя при каждом вызове метода, где такой идентификатор требуется.	21.05.2018	2.30	4.6.17 2017-08-16
Буряков Р.Б.	Исправлены кавычки в примерах API.	31.08.2018	2.31	4.6.17 2017-08-16
Буряков Р.Б.	Исправлены кавычки в примерах API.	02.10.2018	2.32	4.6.17 2017-08-16
Буряков Р.Б.	Добавлен модуль Присвоенные достижения и Изменения игрового счета	09.12.2018	2.33	4.6.35 2018-12-10
Буряков Р.Б.	Обновлено описание типа Файл и раздела Синтаксист вызова. Добавлен модуль Игровой счёт пользователя.	13.06.2019	2.34	4.6.43 2019-06-12
Буряков Р.Б.	Обновлено описание метода Получение содержимого динамического списка. В описание постраничного получения данных добавлены пояснения о максимальном значении limit=200, а также о возвращении всех данных при значении limit большем, чем есть записей в БД.	14.11.2019	2.35	4.6.43 2019-06-12
Буряков Р.Б.	Добавлено описание логического типа полей. Добавлено описание поля pextcode в данных физического лица.	12.08.2020	2.36	4.6.43 2019-06-12
Буряков Р.Б.	Добавлен модуль qua (компетенции)	25.02.2021	2.37	4.6.50 2020-11-06
Буряков Р.Б.	Обновлено описание срока действия сессии API в модуле auth	12.04.2021	2.38	-
Буряков Р.Б.	Добавлен модуль externalRecords (Внешние записи)	17.08.2021	2.39	4.6.43 2021-07-01

Буряков Р.Б.	Исправлена подпись тестовых запросов на подпись, которая получается с ключом «secret»	13.09.2021	2.40	-
Буряков Р.Б.	Добавлен модуль events (События). Добавлен модуль candidates (Кандидаты).	25.04.2022	2.41	4.6.39 2019-11-10
Буряков Р.Б.	Добавлен модуль rubricator (Справочник). Добавлен модуль knowledgeBaseList (База знаний). Добавлен модуль educationDirections (Направления обучения). Добавлен модуль mediausergr (Доступные пользовательские группы ресурсов). Добавлен модуль resources (Доступные ресурсы). Добавлен модуль publications (Доступные публикации). Добавлен модуль usergm (Доступные пользовательские группы мероприятий). Добавлен модуль availableMeasures (Доступные мероприятия). Добавлено описание поля mepasses в модуль measures (Мероприятия). Добавлен модуль mysurvey (Мои опросы).	25.04.2022	2.41	4.6.50 2020-10-21
Буряков Р.Б.	Добавлен метод Добавление преподавателя мероприятия по email. Добавлен модуль measureRegistration (Регистрация на мероприятие).	25.04.2022	2.41	4.6.50 2021-12-10