Инструкция по использованию Amber.SDK v4

Оглавление

Инструкция по использованию Amber.SDK для сайта, разработанного на php.

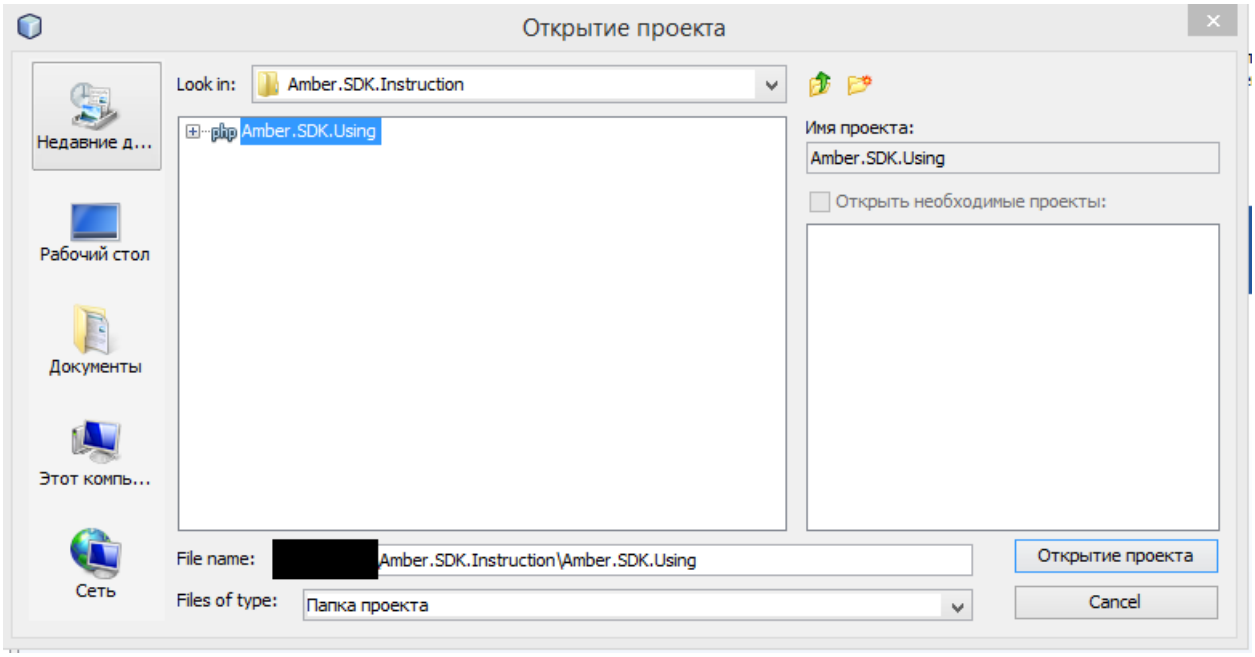
Подключение Amber.SDK к проекту

- Используя composer:

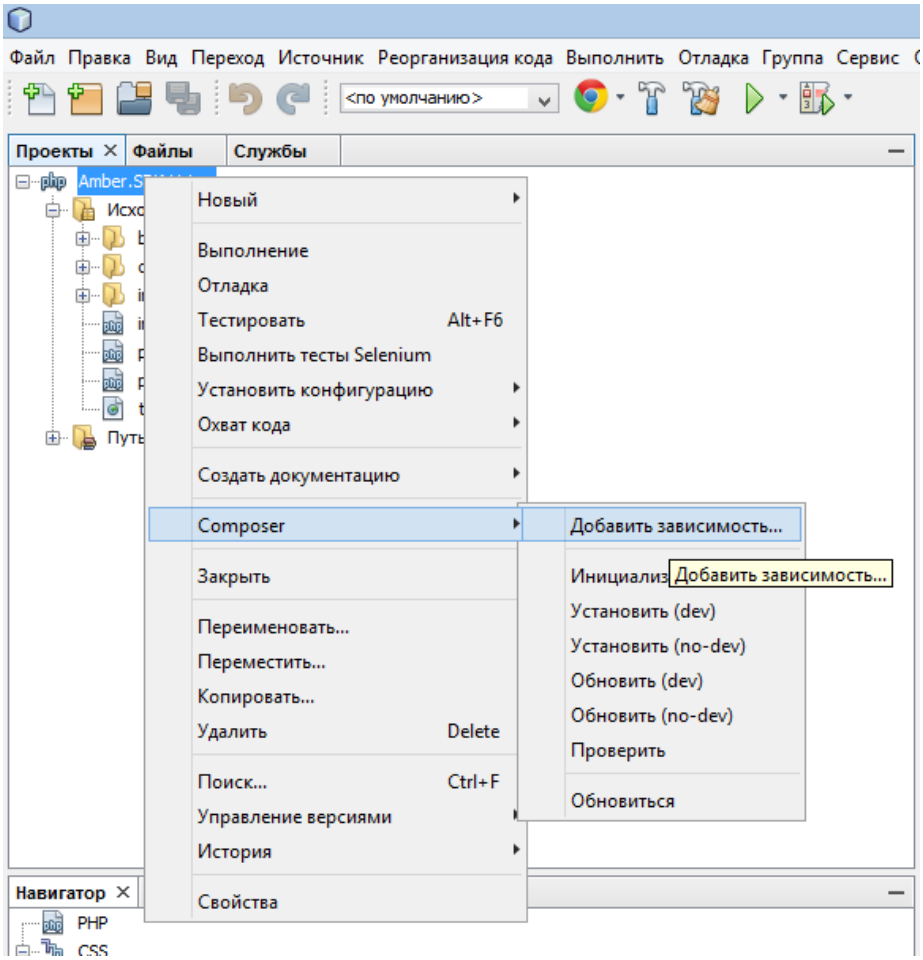
  composer require amber-soft/php.sdk

- В NetBeans – в дереве проекта выбираем «путь к include»и в появившемся окне выбираем путь к SDK.
- Подключение Amber.SDK к php проекту через NetBeans IDE 8.2 + Composer (наиболее предпочтительный):

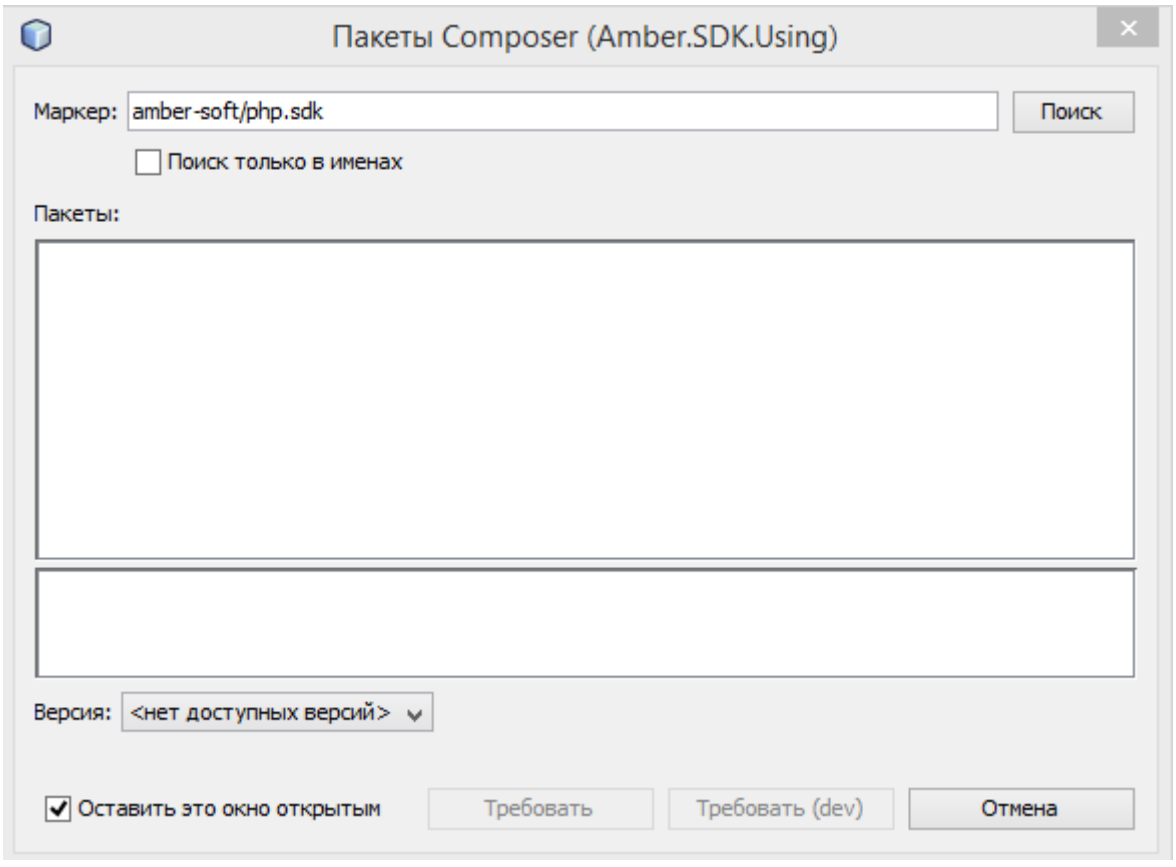
1. Открываем файл проекта Ctrl + Shift + O



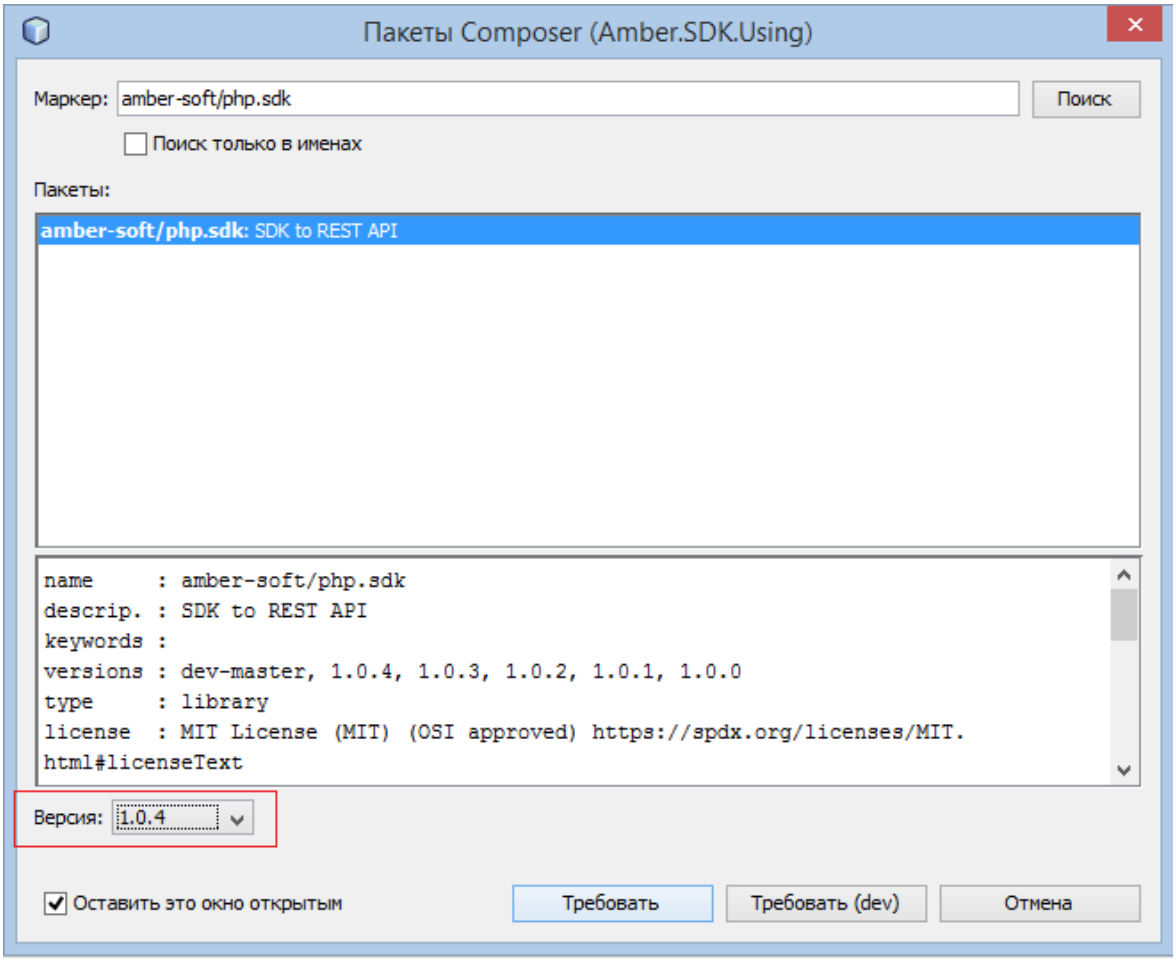
2. Нажимаем Правой кнопкой мыши по проекту и выбираем Composer -> Добавить зависимость…



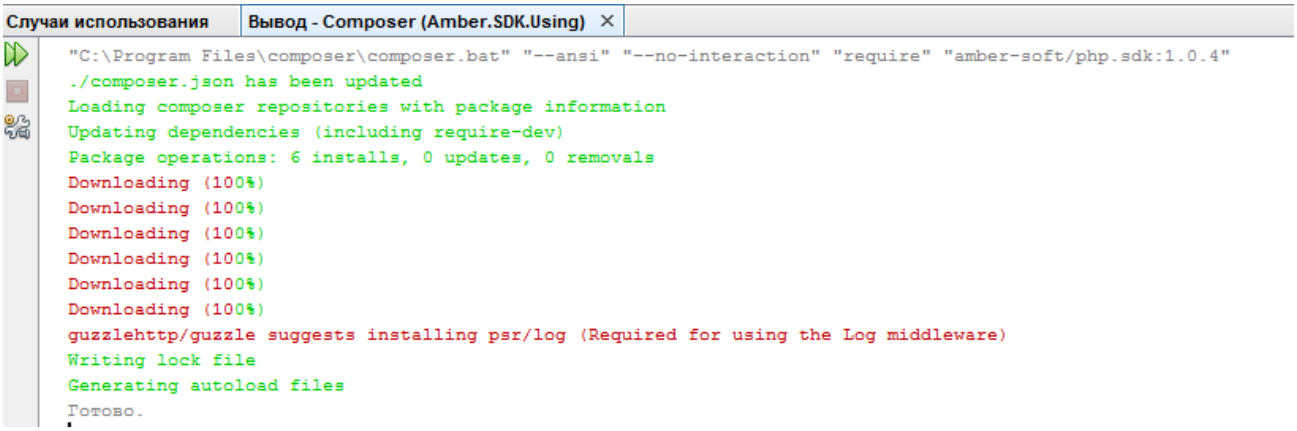
3. В появившемся окне в поле «Маркер» вводим «amber-soft/php.sdk» и выполняем поиск



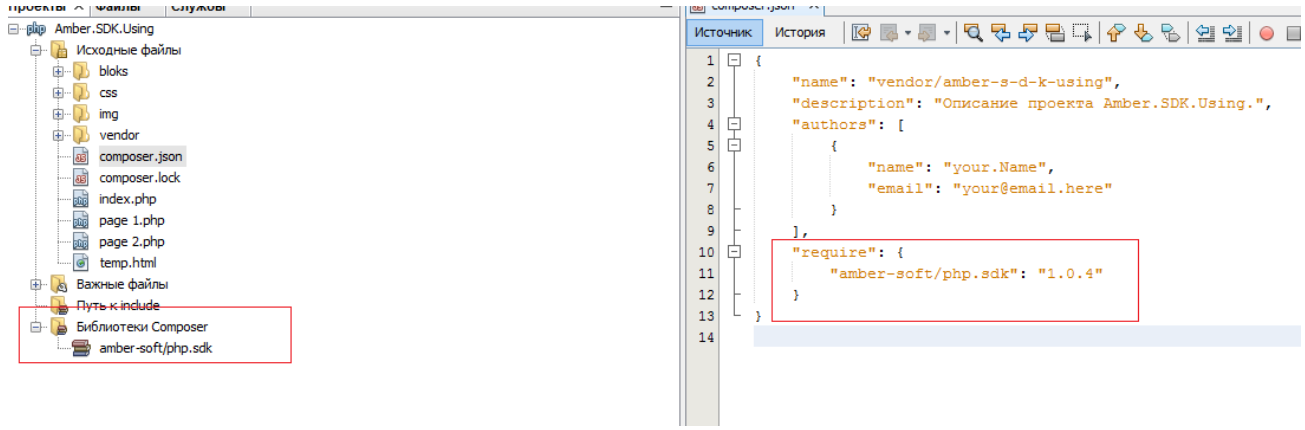
4. В предложенном списке выбираем требуемую для нас версию.



5. Нажимаем «Требовать» (в английской локализации Require), закрываем окно и дожидаемся загрузки.



6. Пакет готов к использованию.
   Данные прописаны в composer.json и библиотека появилась в зависимостях composer.



Конфигурирование и инициация «клиента SDK» для работы с Amber API

1. Для создания подключения нам потребуется конфигурация следующего вида (можно вынести в отдельный файл либо разместить в уже имеющихся конфигурациях, зависит от удобства конкретного потребителя):

```
$config = array(
    'base_uri'=> 'https://your-url.amber-saas.com/', // Хост предоставленного API с учётомпорта
    'endpoint'=>'API/V1.svc/', // URI путь до API сервиса 'user'=>'User1', // Логин пользователя,
    используемый для интеграции'userPassword' => 'userpass', // Пароль пользователя
    'cacheDirectory' => '/tmp/cache', // Директория для хранения кэша 'lock_path' => '/tmp/lock.txt'
);
```

2. С указанной конфигурацией создаём экземпляр класса AppClient:

```
$endpoint = $config['base_uri'] . $config['endpoint'];
$client = new \AmberSdk\Client\AppClient($endpoint, new
\AmberSdk\Client\AuthManager($config));
```

3. Затем используя созданный экземпляр класса, мы можем вызывать необходимые методы для работы с API.

Методы Amber.SDK

Получение записи по идентификатору и коду объекта.

Метод: getObject ($name, $id)

Описание параметров:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$name	string	Код объекта	Да	Отсутствует
$id	int	Идентификатор записи объекта	Да	Отсутствует

Возвращаемые данные:

Тип возвращаемых данных	Описание

object	Возвращает экземпляр объекта со всеми полями и значениями.

Пример запроса:

```
$leadInstance  = $client->getObject (“Leads”, 1);
```

Пример ответа:


```
{Id: "1", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
  Branch: null
  CallID: null
  Comments: null
  CreateBy: "1"
  CreateDate: "2018-01-31T13:55:00.7690009+00:00"
  CustObjVersion: "3747"
  DisqualifyReason: null
  Email: null
  EmailText: null
  Fax: null
  IDOfBranch: null
  IDOfOrg: null
  Id: "1"
  LeadSource: null
  LeadSourceIdentificator: null
  MarketingActivity: null
  MobilePhone: null
  Name: "Тестовый Контрагент / Тестовый контакт"
  Owner: "234"
  PartnerContactName: "Тестовый контакт"
  Phone: null
  PrartnerName: "Тестовый Контрагент"
  ProcessingDate: null
  Source: null
  Specialization: null
  Status: "118"
  UpdateBy: "1"
  UpdateDate: "2018-01-31T13:55:00.7690009+00:00"
  Web: null
  WebRequestNumber: null
  Yuridicheskoenazvanie: null
```


Получение списка экземпляров объекта с возможностью фильтрации и пейджинга.

Метод: getObjects ($name, $filter = [], $size = null, $page = null) (Устаревший, рекомендуется использовать ExecQuery)

Описание параметров:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$name	string	Код объекта	Да	Нет

$filter	array	Ассоциативный массив пар «Поле» => «Значение». Все пары фильтров соединяются условием «ИЛИ» См пример.	Нет	Нет
$size	int	Количество записей на страницу (не может быть изменено)	Нет	50
$page	int	Порядковый номер страницы	Нет	1

Возвращаемые данные:

Тип возвращаемых данных	Описание
array	Массив экземпляров указанного объекта полученных в результате фильтрации и ограничений, переданных в $size и $page

Пример запроса получения 50 записей начиная со второй страницы (в нашем примере на второй странице менее 50 записей):

```
$instances = $client->getObjects("Leads", null, 50, 2);
```

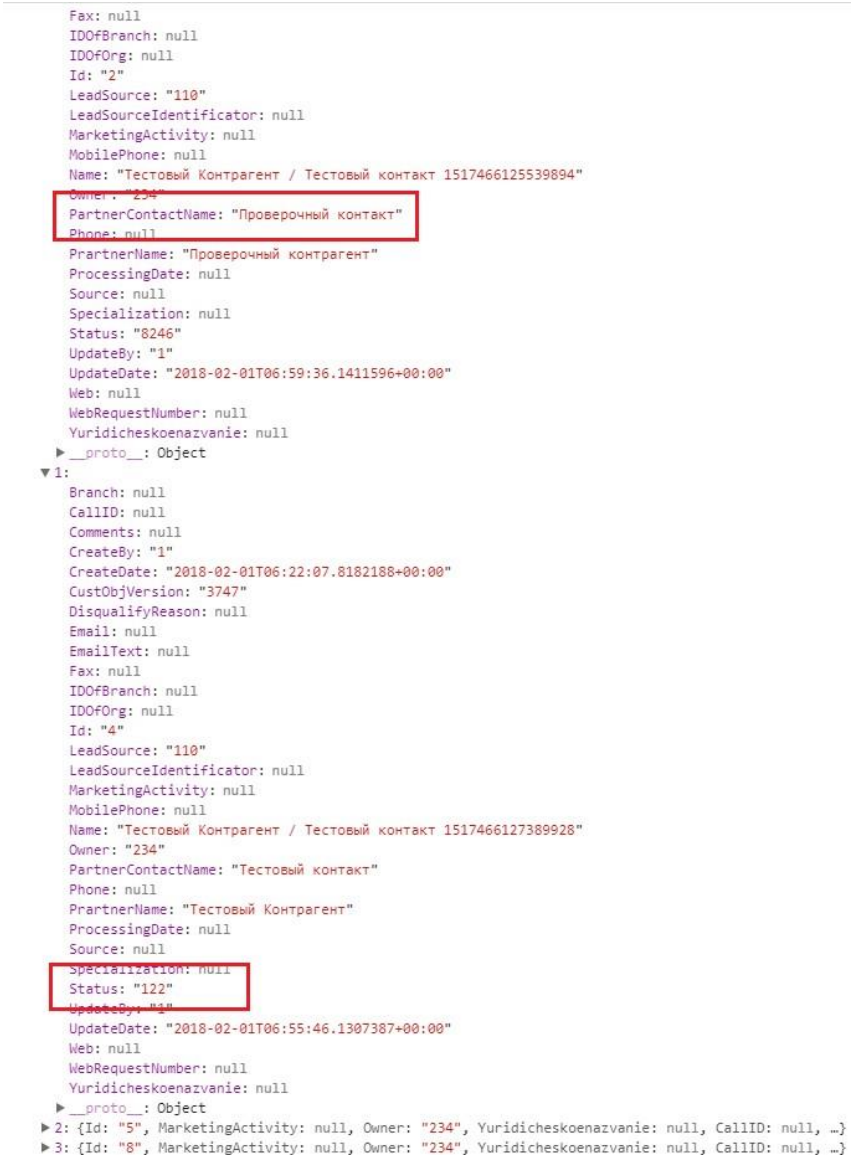
Пример ответа:


```
(7) [{…}, {…}, {…}, {…}, {…}, {…}, {…}]
▶ 0: {Id: "51", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▼ 1:
    Branch: null
    CallID: null
    Comments: null
    CreateBy: "1"
    CreateDate: "2018-02-01T06:22:18.0843092+00:00"
    CustObjVersion: "3747"
    DisqualifyReason: null
    Email: null
    EmailText: null
    Fax: null
    IDOfBranch: null
    IDOfOrg: null
    Id: "52"
    LeadSource: "110"
    LeadSourceIdentificator: null
    MarketingActivity: null
    MobilePhone: null
    Name: "Тестовый Контрагент / Тестовый контакт 1517466137650803"
    Owner: "234"
    PartnerContactName: "Тестовый контакт"
    Phone: null
    PrartnerName: "Тестовый Контрагент"
    ProcessingDate: null
    Source: null
    Specialization: null
    Status: "118"
    UpdateBy: "1"
    UpdateDate: "2018-02-01T06:22:18.0843092+00:00"
    Web: null
    WebRequestNumber: null
    Yuridicheskoenazvanie: null
  ▶ __proto__: Object
▶ 2: {Id: "53", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 3: {Id: "54", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 4: {Id: "55", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 5: {Id: "56", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 6: {Id: "57", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
  length: 7
```


Пример запроса получения первых 50-ти записей объекта с учетом фильтра (в результате примера система выдаёт Записи объекта Leads у которых поле «Контакт» равно «Проверочный контакт» или статус равен «122»):

```
$filter = ['Status' =>   '122', 'PartnerContactName' => 'Проверочный контакт'];
$instances = $client->getObjects("Leads", $filter, 50, 1);
```

Пример ответа:



Создание записи.

Метод: saveObject ($name, $data)

Описание параметров:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$name	string	Код объекта	Да	Отсутствует
$data	array	Ассоциативный массив пар «Поле» => «Значение».	Да	Отсутствует

Возвращаемые данные:

Тип возвращаемых данных	Описание
int	Идентификатор созданного объекта

Пример запроса:

```
$fields = ['Status' =>    '8246', 'Name' => 'Test Lead Name'];
$instances = $client->saveObject("Leads", $fields);
```

Пример ответа (по данному идентификатору можно проверить что всё сохранилось методом getObject:

```
58
```

Результат по getObject:


```
{Id: "58", MarketingActivity: null, Owner: null, Yuridicheskoenazvanie: null, CallID: null, …}
  Branch: null
  CallID: null
  Comments: null
  CreateBy: "4"
  CreateDate: "2018-02-01T07:22:53.4083908+00:00"
  CustObjVersion: "3747"
  DisqualifyReason: null
  Email: null
  EmailText: null
  Fax: null
  IDOfBranch: null
  IDOfOrg: null
  Id: "58"
  LeadSource: null
  LeadSourceIdentificator: null
  MarketingActivity: null
  MobilePhone: null
  Name: "Test Lead Name"
  Owner: null
  PartnerContactName: null
  Phone: null
  PrartnerName: null
  ProcessingDate: null
  Source: null
  Specialization: null
  Status: "8246"
  UpdateBy: "4"
  UpdateDate: "2018-02-01T07:22:53.4083908+00:00"
  Web: null
  WebRequestNumber: null
  Yuridicheskoenazvanie: null
```


Редактирование записи.

Метод: updateObject ($name, $id, $fields)

Описание параметров:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$name	string	Код объекта	Да	Отсутствует
$id	int	Идентификатор изменяемого экземпляра в объекте	Да	Отсутствует
$fields	array	Ассоциативный массив пар «Поле» => «Значение».	Да	Отсутствует

Возвращаемые данные:

Отсутствуют.

Пример запроса:

```
$fields = ['Status' =>   '118', 'Name' => 'Test Lead Name Updated'];
$instances = $client->update Object("Leads", 58, $fields);
```

Пример ответа:

Метод ничего не возвращает. В случае ошибки будут соответствующие эксепшены.

Получение списка записей объекта по расширенным условиям.

Метод: execQuery ($query)

Описание параметров:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$query	ExecutionQuery	Модель ExecutionQuery описывающая условия выборки данных. См описание модели ниже.	Да	Отсутствует

Модель ExecutionQuery:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
customObjectCode	string	Код объекта, по которому делается запрос экземпляров	Да	Отсутствует
select	PropertyPath[]	Загружаемые свойства объекта.	Да	Отсутствует
where	ExpressionElement[]	Условие фильтрации данных.	Нет	Null
orderBy	OrderedPropertyPath[]	Сортировка данных.	Нет	“ID”, “ASC”
offset	Целое	Смещение выборки данных. С какой записи начинать отображаться данные (отсчёт с нуля). Детально рассмотрено в примере.	Нет	0
pageSize	Целое	Размер выборки данных. Максимальное значение – 50.	Нет	50

Модель PropertyPath:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
property	String	Код поля выбираемого объекта. Либо PropertyPath к искомому значению по справочникам через точку. Например, Лид.Название – здесь мы берем поле Лид объекта и у него получаем уже «Название». Так же возможно использовать знак звездочки «*», позволяет получить все поля объекта.	Да	Отсутствует

Модель ExpressionElement:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
brackets	Brackets	Логические скобки	Да	Null
constant	ConstantElement	Константа	Нет	Null
function	FunctionElement	Функция	Нет	Null
operator	Operator	Оператор	Нет	Null
propertyPath	PropertyPath	Свойство объекта, описанное моделью PropertyPath.	Нет	0

Модель Brackets:

Название	Тип	Описание	Обязательный?	Значение по умолчанию

$items	array	Массив ExpressionElement представляющий собой условие в скобках	Да	Отсутствует

Модель ConstantElement:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$dataTypeCode	String	Код типа данных константы. См Приложение 1, Таблица 1 (Допустимые типы данных)	Да	Отсутствует
$value	String	Значение константы	Да	Отсутствует

Модель Operator:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$operatorCode	String	Код Оператора. См Приложение 1, Таблица 2 (Допустимые операторы)	Да	Отсутствует

Модель FunctionElement:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$functionCode	string	Код функции. См. Приложение 1, Таблица 3 (Допустимые системные функции).	Да	Отсутствует
$value	ExpressionElement[]	Массив аргументов, передаваемый в функцию.	Да	Отсутствует

Модель OrderedPropertyPath:

Название	Тип	Описание	Обязательный?	Значение по умолчанию
$property	string	Код свойства, по которому следует производить сортировку	Да	Отсутствует
$order	string	Направление сортировки	Да	Отсутствует

Возвращаемые данные:

Тип возвращаемых данных	Описание
array	Массив экземпляров указанного объекта полученных в результате исполнения $query

Пример 1: Запрос на получение первых 40-ка записей в объекте Leads (без фильтрации):

```
$select = [
        new \AmberSdk\Client\Model\PropertyPath("*")
];
$query = new \AmberSdk\Client\Model\ExecutionQuery();
$query->select=$select;
$query->customObjectCode= "Leads";
$query ->where = null;
$query->pageSize = 40;
$query->offset = 0;
$leads = $client->execQuery($query);
```

Ответ:

```
▶ 19: {Id: "20", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 20: {Id: "21", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 21: {Id: "22", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 22: {Id: "23", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 23: {Id: "24", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 24: {Id: "25", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 25: {Id: "26", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 26: {Id: "27", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 27: {Id: "28", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 28: {Id: "29", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 29: {Id: "30", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 30: {Id: "31", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 31: {Id: "32", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▼ 32:
    Branch: null
    CallID: null
    Comments: null
    CreateBy: "1"
    CreateDate: "2018-02-01T06:22:13.8966191+00:00"
    CustObjVersion: "3747"
    DisqualifyReason: null
    Email: null
    EmailText: null
    Fax: null
    IDOfBranch: null
    IDOfOrg: null
    Id: "33"
    LeadSource: "110"
    LeadSourceIdentificator: null
    MarketingActivity: null
    MobilePhone: null
    Name: "Тестовый Контрагент / Тестовый контакт 1517466133466333"
    Owner: "234"
    PartnerContactName: "Тестовый контакт"
    Phone: null
    PrartnerName: "Тестовый Контрагент"
    ProcessingDate: null
    Source: null
    Specialization: null
    Status: "118"
    UpdateBy: "1"
    UpdateDate: "2018-02-01T06:22:13.8966191+00:00"
    Web: null
    WebRequestNumber: null
    Yuridicheskoenazvanie: null
  ▶ __proto__: Object
▶ 33: {Id: "34", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 34: {Id: "35", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 35: {Id: "36", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 36: {Id: "37", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 37: {Id: "38", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 38: {Id: "39", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
▶ 39: {Id: "40", MarketingActivity: null, Owner: "234", Yuridicheskoenazvanie: null, CallID: null, …}
  length: 40
```


Пример 2: Запрос с выбором поля «Название» у 10 записей начиная с 10-ой + фильтрация по названию статуса:

```
$select = [
        new \AmberSdk\Client\Model\PropertyPath("Name")
];
$query = new \AmberSdk\Client\Model\ExecutionQuery();
$query->select=$select;
$query->customObjectCode= "Leads";

$property1 = new \AmberSdk\Client\Model\ExpressionElement();
$property1->propertyPath = new \AmberSdk\Client\Model\PropertyPath("Status.Name");
$operatorEqual = new \AmberSdk\Client\Model\ExpressionElement();
$operatorEqual->operator = new \AmberSdk\Client\Model\Operator("EqualTo");
$constantStatus = new \AmberSdk\Client\Model\ExpressionElement();
$constantStatus->constant = new \AmberSdk\Client\Model\ConstantElement("String", "Новый");
$Condition1 = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$property1, $operatorEqual, $constantStatus]));
$query->where = [$Condition1];
$query->pageSize = 10;
$query->offset = 9;
$leads = $client->execQuery($query);
```

Ответ:


```
(10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
▶ 0: {Name: "Тестовый Контрагент / Тестовый контакт 1517466129993841"}
▶ 1: {Name: "Тестовый Контрагент / Тестовый контакт 1517466130176636"}
▶ 2: {Name: "Тестовый Контрагент / Тестовый контакт 1517466130404148"}
▶ 3: {Name: "Тестовый Контрагент / Тестовый контакт 1517466130608659"}
▶ 4: {Name: "Тестовый Контрагент / Тестовый контакт 1517466130809428"}
▶ 5: {Name: "Тестовый Контрагент / Тестовый контакт 151746613097515"}
▶ 6: {Name: "Тестовый Контрагент / Тестовый контакт 1517466131171940"}
▶ 7: {Name: "Тестовый Контрагент / Тестовый контакт 1517466131349875"}
▶ 8: {Name: "Тестовый Контрагент / Тестовый контакт 1517466131536396"}
▶ 9: {Name: "Тестовый Контрагент / Тестовый контакт 1517466131735195"}
  length: 10
```

Пример 3: Запрос записей объекта по двум условиями фильтрации связанных по условию «И» (Статус равно «Новый» И Дата обработки «заполнено»):

```
$select = [
        new \AmberSdk\Client\Model\PropertyPath("Name"),
        new \AmberSdk\Client\Model\PropertyPath("ProcessingDate")
];
$query = new \AmberSdk\Client\Model\ExecutionQuery();
$query->select=$select;
$query->customObjectCode= "Leads";
$property1 = new \AmberSdk\Client\Model\ExpressionElement();
$property1->propertyPath = new \AmberSdk\Client\Model\PropertyPath("Status.Name");
$operatorEqual = new \AmberSdk\Client\Model\ExpressionElement();
$operatorEqual->operator = new \AmberSdk\Client\Model\Operator("EqualTo");

$constantStatus = new \AmberSdk\Client\Model\ExpressionElement();
$constantStatus->constant = new \AmberSdk\Client\Model\ConstantElement("String", "Новый");
$Condition1 = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$property1, $operatorEqual, $constantStatus]));

$operatorConditionsOr = new \AmberSdk\Client\Model\ExpressionElement();
$operatorConditionsOr->operator = new \AmberSdk\Client\Model\Operator("And");//OR или AND??

$property2 = new \AmberSdk\Client\Model\ExpressionElement();
$property2->propertyPath = new \AmberSdk\Client\Model\PropertyPath("ProcessingDate");
$operatorIsNotNull = new \AmberSdk\Client\Model\ExpressionElement();
$operatorIsNotNull->operator = new \AmberSdk\Client\Model\Operator("IsNotNull");
$Condition2 = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$property2, $operatorIsNotNull]));

$bracketsConditions = new
\AmberSdk\Client\Model\Brackets([$Condition1, $operatorConditionsOr, $Condition2 ]);

$predicat = new \AmberSdk\Client\Model\ExpressionElement($bracketsConditions);

$query->where = [$predicat];
$query->pageSize = 10;
$query->offset = 0;
$leads = $client->execQuery($query);
```

Ответ:

```
(2) [{…}, {…}]
▶ 0: {Name: "Тестовый Контрагент / Тестовый контакт 1517466128608504", ProcessingDate: "2018-02-15T00:00:00+00:00"}
▶ 1: {Name: "Тестовый Контрагент / Тестовый контакт 1517466128782523", ProcessingDate: "2018-02-14T00:00:00+00:00"}
  length: 2
```

Пример 4: «Пример 3» плюс к результату добавляется сортировка (в начале по ProcessingDate, затем по Name):

```
$select = [
        new \AmberSdk\Client\Model\PropertyPath("Name"),
        new \AmberSdk\Client\Model\PropertyPath("ProcessingDate")
];
$query = new \AmberSdk\Client\Model\ExecutionQuery();
$query->select=$select;
$query->customObjectCode= "Leads";
$property1 = new \AmberSdk\Client\Model\ExpressionElement();
$property1->propertyPath = new \AmberSdk\Client\Model\PropertyPath("Status.Name");
$operatorEqual = new \AmberSdk\Client\Model\ExpressionElement();
$operatorEqual->operator = new \AmberSdk\Client\Model\Operator("EqualTo");

$constantStatus = new \AmberSdk\Client\Model\ExpressionElement();
$constantStatus->constant = new \AmberSdk\Client\Model\ConstantElement("String", "Новый");
$Condition1 = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$property1, $operatorEqual, $constantStatus]));

$operatorConditionsOr = new \AmberSdk\Client\Model\ExpressionElement();
$operatorConditionsOr->operator = new \AmberSdk\Client\Model\Operator("And");

$property2 = new \AmberSdk\Client\Model\ExpressionElement();
$property2->propertyPath = new \AmberSdk\Client\Model\PropertyPath("ProcessingDate");
$operatorIsNotNull = new \AmberSdk\Client\Model\ExpressionElement();
$operatorIsNotNull->operator = new \AmberSdk\Client\Model\Operator("IsNotNull");
$Condition2 = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$property2, $operatorIsNotNull]));
```

```
$bracketsConditions = new
\AmberSdk\Client\Model\Brackets([$Condition1, $operatorConditionsOr, $Condition2 ]);

$predicat = new \AmberSdk\Client\Model\ExpressionElement($bracketsConditions);

$orderedByName =   new \AmberSdk\Client\Model\OrderedPropertyPath("Name", "DESC");
$orderedByProcessingName =   new \AmberSdk\Client\Model\OrderedPropertyPath("ProcessingDate", "ASC");
```

```
$query->where = [$predicat];
$query->pageSize = 10;
$query->offset = 0;

$query->orderBy = [$orderedByProcessingName, $orderedByName];
$leads = $client->execQuery($query);
```

Ответ:

```
(2) [{…}, {…}]
▶ 0: {Name: "Тестовый Контрагент / Тестовый контакт 1517466128782523", ProcessingDate: "2018-02-14T00:00:00+00:00"}
▶ 1: {Name: "Тестовый Контрагент / Тестовый контакт 1517466128608504", ProcessingDate: "2018-02-15T00:00:00+00:00"}
  length: 2
```

Пример 5: Использование системных функций в условиях фильтрации (пример показывает условие выбора записей: «Если (1 = 1), то (Id >= 2), иначе (Owner >= 2)»)

```
$select = [
        new \AmberSdk\Client\Model\PropertyPath("Id"),
        new \AmberSdk\Client\Model\PropertyPath("Name"),
        new \AmberSdk\Client\Model\PropertyPath("ProcessingDate")
];
$query = new \AmberSdk\Client\Model\ExecutionQuery();
$query->select=$select;
$query->customObjectCode= "Leads";


$Argument1 = new \AmberSdk\Client\Model\ExpressionElement();
$Argument1->constant = new \AmberSdk\Client\Model\ConstantElement("Int", "1");
$operatorEqual1 = new \AmberSdk\Client\Model\ExpressionElement();
$operatorEqual1->operator = new \AmberSdk\Client\Model\Operator("EqualTo");
$Argument2 = new \AmberSdk\Client\Model\ExpressionElement();
$Argument2->constant = new \AmberSdk\Client\Model\ConstantElement("Int", "1");
$ArgumentsExpression = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$Argument1, $operatorEqual1, $Argument2]));

$resultProperty1 = new \AmberSdk\Client\Model\ExpressionElement();
$resultProperty1->propertyPath = new \AmberSdk\Client\Model\PropertyPath("Id");
$resultProperty2 = new \AmberSdk\Client\Model\ExpressionElement();
$resultProperty2->propertyPath = new \AmberSdk\Client\Model\PropertyPath("Owner");

$functionExpression = new \AmberSdk\Client\Model\ExpressionElement();
$functionExpression->function = new
\AmberSdk\Client\Model\FunctionElement("IIF", [$ArgumentsExpression, $resultProperty1, $resultProperty2 ] );;

$operatorEqual = new \AmberSdk\Client\Model\ExpressionElement();
$operatorEqual->operator = new \AmberSdk\Client\Model\Operator("GreaterThanOrEqualTo");

$constant4 = new \AmberSdk\Client\Model\ExpressionElement();
$constant4->constant = new \AmberSdk\Client\Model\ConstantElement("Int", "2");

$FinalExpression = new \AmberSdk\Client\Model\ExpressionElement(new
\AmberSdk\Client\Model\Brackets([$functionExpression, $operatorEqual, $constant4]));

$query->where = [$FinalExpression];
$query->pageSize = 10;
$query->offset = 0;
$leads = $client->execQuery($query);
```

Ответ:

```
(10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
▶ 0: {Id: "2", Name: "Тестовый Контрагент / Тестовый контакт 1517466125539894", ProcessingDate: null}
▶ 1: {Id: "3", Name: "Тестовый Контрагент / Тестовый контакт 1517466126708259", ProcessingDate: null}
▶ 2: {Id: "4", Name: "Тестовый Контрагент / Тестовый контакт 1517466127389928", ProcessingDate: null}
▶ 3: {Id: "5", Name: "Тестовый Контрагент / Тестовый контакт 1517466127640510", ProcessingDate: "2018-01-29T00:00:00+00:00"}
▶ 4: {Id: "6", Name: "Тестовый Контрагент / Тестовый контакт 1517466128005193", ProcessingDate: null}
▶ 5: {Id: "7", Name: "Тестовый Контрагент / Тестовый контакт 1517466128204785", ProcessingDate: null}
▶ 6: {Id: "8", Name: "Тестовый Контрагент / Тестовый контакт 1517466128371461", ProcessingDate: null}
▶ 7: {Id: "9", Name: "Тестовый Контрагент / Тестовый контакт 1517466128608504", ProcessingDate: "2018-02-15T00:00:00+00:00"}
▶ 8: {Id: "10", Name: "Тестовый Контрагент / Тестовый контакт 1517466128782523", ProcessingDate: "2018-02-14T00:00:00+00:00"}
▶ 9: {Id: "11", Name: "Тестовый Контрагент / Тестовый контакт 1517466129015187", ProcessingDate: null}
  length: 10
```

Приложение 1.

Таблица 1. Допустимые типы данных.

dataTypeCode
Int
Bit
String
LongString
UniqueIdentifier
Decimal
Money
LongDateTime
LongDateTime
Time
BigInt
LongString
String
String
Decimal
Fileinfo
ImageInfo

Таблица 2. Допустимые коды операторов.

operatorCode	Описание
EqualTo	Равно
GreaterThan	Больше
GreaterThanOrEqualTo	Больше или равно
LessThan	Меньше
LessThanOrEqualTo	Меньше или равно
NotEqualTo	Не равно
Plus	Плюс
Minus	Минус
Multiply	Умножить
Divide	Разделить
Modulo	Остаток от деления
And	Логическое И
Or	Логическое ИЛИ
IsNull	Не заполнено Формат: <Операнд> IsNull;
IsNotNull	Заполнено Формат: <Операнд> IsNotNull;
Not	Логическое отрицание Формат: Not <Операнд>;
MinusUnary	Унарный минус Формат: MinusUnary <Операнд>;
StartsWith	Строка начинается на значение
EndsWith	Строка заканчивается на значение
Contains	Строка содержит значение
OnDate	На дату

Таблица 3. Доступные системные функции.

functionCode	Описание

IIF	Условие. Функция проверяет условие, записанное в первом аргументе. Если условие выполняется - возвращает значение второго аргумента. Если не выполняется - значение третьего аргумента;
SYSDATETIMEOFFSET	Текущее системное время UTC.