I. Подготовительные работы по интеграции с применением API

Доступ к описанию методов API

Базовое описание см. в:

- методы и типы регистрации/обновления сведений о подключении сервиса: **https://sandbox.goodfin.ru/docs/shb-open-api/v1/index.html**

- методы и типы описания бизнес-процесса отправки заявок в сервисы и обмена сообщениями с сервисом: **https://sandbox.goodfin.ru/docs/bl-open-api/v1/index.html**

Концепция организации передаваемых данных, используемая в API

Примеры сформированных тестовых заявок на дату 30/12/2020 со структурой, актуальной для версии 1.19 и выше, по которым можно посмотреть структуру данных:

для типа продукта Банковская гарантия на исполнение от клиента ЮЛ - БГ(И) ЮЛ.json

для типа продукта Банковская гарантия на исполнение от клиента ИП - БГ(И) ИП.json

для типа продукта Банковская гарантия на возврат аванса от клиента ЮЛ - БГ(А) ЮЛ.json

для типа продукта Банковская гарантия обеспечения гарантийных обязательств от клиента ЮЛ - БГ(О) ЮЛ.json

для типа продукта Банковская гарантия на участие от клиента ЮЛ - БГ(У) ЮЛ.json

для типа продукта Тендерный займ от клиента ЮЛ - ТЗ ЮЛ.json

для типа продукта Кредит на исполнение контракта от клиента ЮЛ - КиК ЮЛ.json

1. Нормализованная форма данных

В некоторых методах API применена нормализация данных.

В нормализованном виде взаимосвязь между объектами определяется явными ссылками, а сами объекты с данными лежат в специальной области структуры - fetchFields.

В структуре блока fetchFields объекты распределены по их типу. Общий формат передаваемых данных можно определить следующим образом:

{

```
  "result": {
    "referenceToObject1": "obj1",
    ...
  },
  "fetchFields": {
    "Object1": {
      "obj1": {
        "id": "obj1",
        "referenceToObject2": "obj2",
        ...
      }
    },
    "Object2": {
      "obj2": {
        "id": "obj2",
        ...
      }
    },
    ...
  }
}
```

2. Концепция абстрактной ссылки

Если на уровне структуры данных поле ссылается на объект некой абстракции, то данное поле применяет концепцию абстрактной ссылки. Сама ссылка определяет какого типа объект по его идентификатору, указанному после разделителя.

В следующем примере концепция абстрактной ссылки была применена к полю "result.referenceToAbstractObject" . Его содержимое - "ConcreteObject1:obj1" указывает на то, что данные объекта необходимо искать по следующему пути: "fetchFields.ConcreteObject1.obj1". Варианты типов ссылки ограничены типом самой ссылки.

```
{
  "result": {
    "referenceToAbstractObject": ["ConcreteObject1:obj1", "ConcreteObject2:obj2"]
  },
  "fetchFields": {
    "ConcreteObject1": {
      "obj1": {
        "id": "obj1",
        "dataField1": "data111"
      }
    },
    "ConcreteObject2": {
      "obj2": {
        "id": "obj2",
        "dataField2": "data222"
      }
    }
  }
}
```

Концепция абстрактной ссылки также применяется для обобщающих проекций. В примере ниже такой вариант показан в "AbstractProjection".

У данного вида объектов идентифицирующим полем является абстрактная ссылка. В остальном обобщающая проекция ничем не отличается от других объектов.

```
{
  "result": {
    "referenceToAbstractObject": ["ConcreteObject1:obj1", "ConcreteObject2:obj2"]
  },
  "fetchFields": {
    "AbstractProjection": {
      "ConcreteObject1:obj1": {
        "id": "ConcreteObject1:obj1",
        "fld": "43434035848"
      },
      "ConcreteObject2:obj2": {
        "id": "ConcreteObject2:obj2",
        "fld": "54545467676"
      }
    }
  }
}
```

В случаях, когда у объекта есть данные, которые требуется подгружать опционально, используются дополнительные объекты-проекции с дополнительными данными, при этом идентификатор равен идентификатору основного объекта.

```
{
```

```
  "result": {
    "referenceToConcreteObject": "obj1"
  },
  "fetchFields": {
    "ConcreteObject1": {
      "obj1": {
        "id": "obj1",
        "dataField1": "data111"
      }
    },
    "AdditionalDataForConcreteObject1": {
      "obj1": {
        "id": "obj1",
        "additionalData": "vfgsdljkg987ihdfj"
      }
    }
  }
}
```

3. Концепция идентификации - композитный идентификатор

Для композиции объектов используется также отдельная концепция идентификации - композитный идентификатор.

Объекты, которые являются частью композиции, в своем идентификаторе всегда имеют идентификатор своего композита. Таким образом, в примере ниже идентификатор "comp1:item1" имеет в своем содержимом и идентификатор своего композита "comp1".

Это дает гарантию, что при пакетной обработке объектов, которые являются частью композита, можно всегда выйти на сам объект композиции. Это дает возможность выбирать все части композиции, фильтруя объекты по их идентификатору функцией *startWith*("comp1") .

Несмотря на то, что сейчас для многих идентификаторов, в том числе и композитных, используется uuid, уникальность идентификатора "CompositeObjectItem" гарантируется только в пределах его композита("comp1").

```
{
  "result": {
    "referenceToObject": "comp1"
  },
  "fetchFields": {
    "CompositeObject": {
      "comp1": {
        "id": "comp1",
        "dataField3": "data123"
      }
    },
    "CompositeObjectItem": {
      "comp1:item1": {
        "id": "comp1:item1",
        "dataField4": "data321"
      },
      "comp1:item2": {
        "id": "comp1:item2",
        "dataField4": "321data"
      }
    }
  }
}
```

В "fetchFields" так же могут возвращаться объекты объедененные общей абстракцией. В этом случае их конкретный тип зависит от одного из полей, который присутвует во всех конкретных типах. На примере ниже таким полем является "type". В зависимости от значения данного поля, будет определено, какие дополнительные поля применимы к данному объекту.

```
{
  "result": {
    "referenceToObject": "obj1"
  },
  "fetchFields": {
    "AbstractObject": {
      "obj1": {
        "id": "obj1",
        "type": "type1",
        "fldType1": 2323
      },
      "obj2": {
        "id": "obj2",
        "type": "type2",
        "fldType2": "djfklsfhjf"
      },
      "obj3": {
        "id": "obj3",
        "type": "type2",
        "fldType2": "ppisaiosifg"
      }
    }
  }
}
```


4. Пример

Ниже приведен пример данных заявки в нормализованном виде, часть данных опущена, чтобы сократить объем.

В данном примере поле "result.client" применяет концепцию абстрактной ссылки, при этом тип данной ссылки позволяет также ссылаться на объекты типа "IndividualContactParticipant".

Объект "Person" является частью композита "Company". Поля "author" и "modifier" объекта "Company" разыменовываются через проекцию "ModifierParticipant".


```
{
  "result": {
    "id": "e452c6e2-449a-4f4b-8da2-dfaecc194e0b",
    "dealNumber": "()-6648",
    "currencyType": "1",
    "income": 7206.00,
    "client": "Company:32a45baf-ef78-4459-96a8-1d133cf2b913"
  },
  "fetchFields": {
    "Person": {
      "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959": {
        "id": "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959",
        "firstName": "",
        "middleName": "",
        "docSeries": "4607",
        "docType": "PASSPORT",
      }
    },
    "OktmoCatalog": {
      "23129": {
        "id": "23129",
        "parentId": "23128",
        "code": "17701000001",
        "name": " "
      }
    },
    "Company": {
      "32a45baf-ef78-4459-96a8-1d133cf2b913": {
        "id": "32a45baf-ef78-4459-96a8-1d133cf2b913",
        "oktmo": "23129",
        "legalAddress": "32a45baf-ef78-4459-96a8-1d133cf2b913:0c8ecd6f-d7f7-41d7-b15a-ed850dc24944",
        "author": "IndividualContactParticipant:961b4042-a55d-4b7e-8edf-33da2884e33c",
        "createDateTime": "2019-02-04T14:52:59.865",
        "modifier": "CompanyContactParticipant:29bb5e27-8f2f-4001-afa2-61292ab971eb",
        "modifiedDateTime": "2019-08-02T13:14:28.084"
      }
    },
    "PersonRole": {
      "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959:5f5869a4-e6b2-4a5f-b254-
9583689954b3": {
        "id": "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959:5f5869a4-e6b2-4a5f-
b254-9583689954b3",
        "roleType": "BENEFICIAR",
```

```
        "percent": 100.0000,
        "date": null
      },
      "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959:f708ce68-386a-445c-87ea-
18c826ca05e6": {
        "id": "32a45baf-ef78-4459-96a8-1d133cf2b913:92bd540a-3ce5-4a51-b1a1-5e9bf085f959:f708ce68-386a-445c-
87ea-18c826ca05e6",
        "roleType": "BOOKER",
        "fullPostName": " ",
        "postDate": null,
        "endDate": null,
        "workYears": null,
        "workPrev": null
      }
    },
    "CurrencyTypeCatalog": {
      "1": {
        "id": "1",
        "code": "RUB",
        "name": " ",
        "codeNum": "643"
      }
    },
    "ModifierParticipant": {
      "CompanyContactParticipant:29bb5e27-8f2f-4001-afa2-61292ab971eb": {
        "id": "CompanyContactParticipant:29bb5e27-8f2f-4001-afa2-61292ab971eb",
        "firstName": "",
        "middleName": ""
      },
      "IndividualContactParticipant:961b4042-a55d-4b7e-8edf-33da2884e33c": {
        "id": "IndividualContactParticipant:961b4042-a55d-4b7e-8edf-33da2884e33c",
        "firstName": "",
        "middleName": ""
      }

    },
    "CompanyContactParticipant": {
      "29bb5e27-8f2f-4001-afa2-61292ab971eb": {
        "id": "29bb5e27-8f2f-4001-afa2-61292ab971eb",
        "firstName": "",
        "middleName": "",
        "userEmail": "alex@example.com",
        "company": "32a45baf-ef78-4459-96a8-1d133cf2b913",
        "verificationType": "ECP"
      }
    }
  }
}
```


Примерный порядок работ для подготовки к интеграции с Goodfin с использованием песочницы