Jade

Панов Никита Б05-033

История

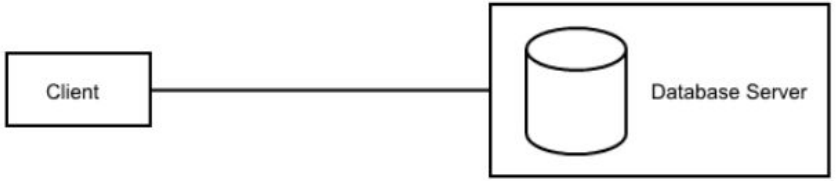
- Первый релиз - 1996.
- Разработана в Новой Зеландии, Jade Software Corporation.
- Целая экосистема - язык программирования, IDE, debugger, клиент и сервер.
- Последний релиз - JADE 2022 SP1.
- Сайт - https://www.jadeworld.com/jade-platform/latest-platform (не любят Россию, но с правильным VPN все работает).
- Там же лежат гайды по установке и использованию.
- Для использование требуется получить лицензию, получение студенческой лицензии работает.
- Существует API для использования из других ЯП (Java, C++)

Немного информации

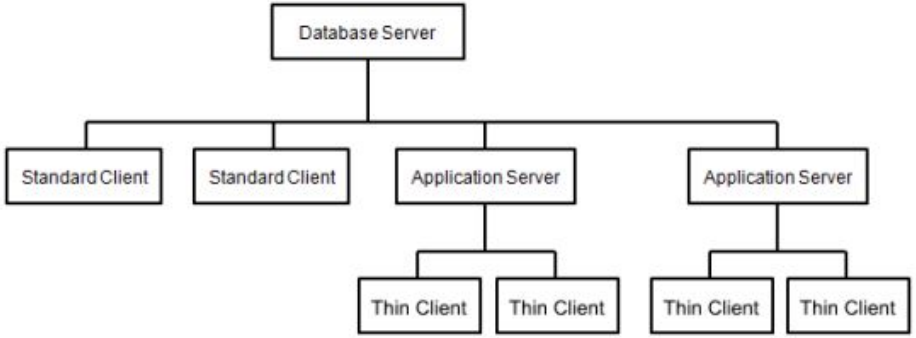
- Все это дело проприетарное, написано судя по всему на C++.
- Объектно-ориентированная база данных.
- https://www.jadeworld.com/jade-platform/developer-centre/jug - лежат доклады с идеями насчет дальнейшего развития и оптимизаций.
- https://secure.jadeworld.com/JADETech/JADE2022/OnlineDocumentation/ - онлайн документация.
- https://secure.jadeworld.com/JADETech/Education/InstallAdminCourse/JadeInstallAdminCourse.pdf - для администрирования.
- https://secure.jadeworld.com/JADETech/Education/DevCourse/JadeDevCourse.pdf - для разработчиков.
- https://secure.jadeworld.com/JADETech/JADE2016/WhitePapers/WP_SDS.pdf - распределенный database server.

Как развернуть локально

- С официального сайта можно поставить только под windows.
- После установки появляются куча приложений, нас интересуют следующие:
  a. JADE - single user режим, все запросы и действия доступны только одном процессу из под запущенной IDE.
  b. JADE Client и JADE Database Server - multi user режим, запускается сервер, после чего к нему подключаются клиенты



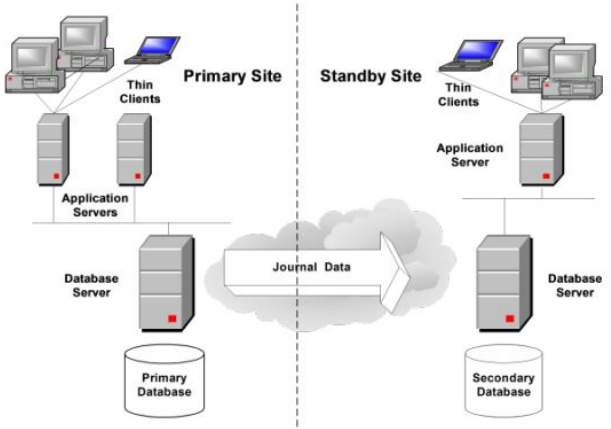
Архитектура



- Database Server - основной сервер на работе которого завязана работа всей системы.
- Standard client - клиент, работающий напрямую с сервером.
- Application Server - поддерживает соединение с клиентами и основным сервером. Требуется хорошее LAN соединение между Database Server и Application Server.
- На серверах и клиентах существуют Persistent и Transient caches.
- Поддерживаются периодические бэкапы и ведение журналов транзакций. В случае аварии, после рестарта, данные автоматически будут восстановлены.

Synchronized Database Service

- Распределенный Database Server.
- Периодически пересылается журнал транзакций.
- Вторичный сервер может принимать подключения от клиентов, а может быть пассивным.



Security

- Шифрование бэкапов.
- Шифрование самих данных.
- Расшифровкой занимается клиент, пересылаются данные в зашифрованном виде.
- Возможна настройка какие объекты шифровать, а какие - нет.

Язык запросов

- По ощущениям - смесь Pascal и Java. При создании новых объектов, они могут быть добавлены на Database server.
- Созданные объекты можно хранить в коллекциях, и получать их из них, либо, существует метод который позволяет получить все объекты данного класса (instances).
- Существуют классы, интерфейсы, полиморфизм и наследование.
- По умолчанию, запросы выполняются преимущественно на клиентской машине, однако, например для работы с большим количеством данных, можно указать чтобы код выполнялся на сервере.

```
calledMethod01(parameters): returnType serverExecution;
```

(Честно, IDE очень неудобна и сам язык выглядит невероятно странно)

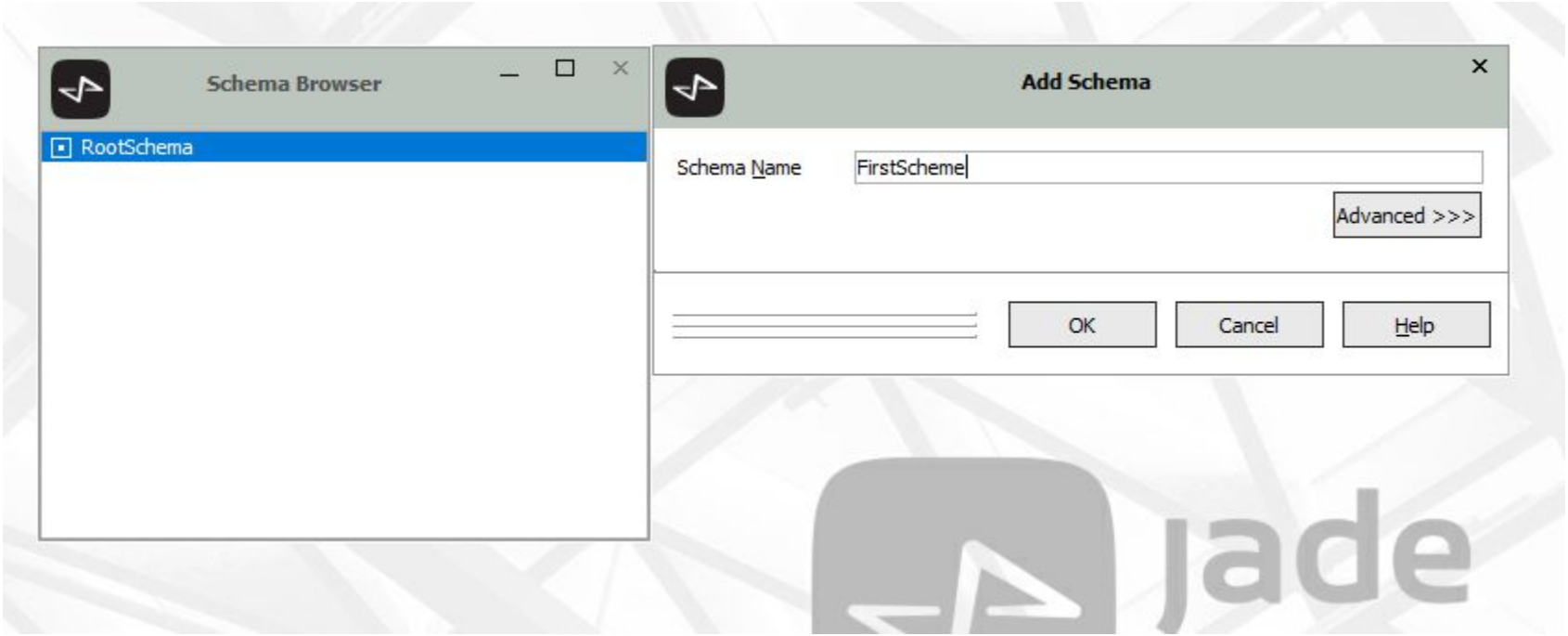
Коллекции

- Array - индексируемый массив.
- Dictionary - key-value коллекция. Ключ может быть как полем объекта, так и внешним.
- Set - неупорядоченное множество.
- Также существуют итераторы, позволяющие итерироваться по коллекциям.

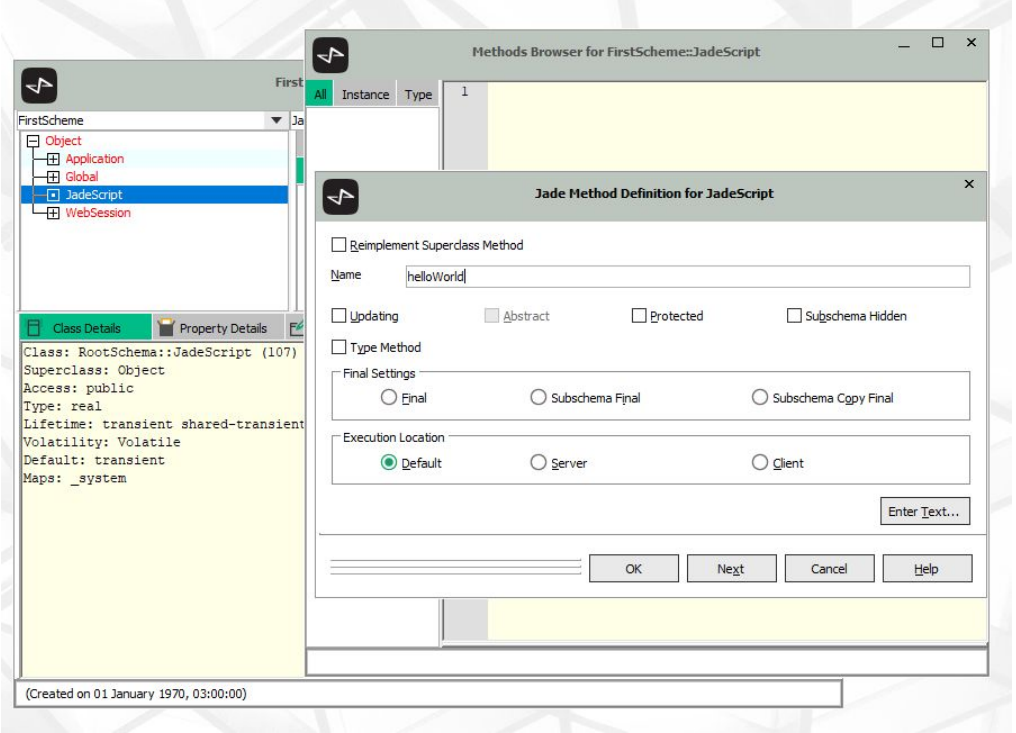
```
iter := coll.createIterator();
while iter.next(cust) do
    write cust.lastName;
endwhile;
delete iter;
```

```
iter := coll.createIterator();
coll.startKeyGeq("Jones", iter);
while iter.next(cust) do
    write cust.lastName;
endwhile;
delete iter;
```

Создание схемы



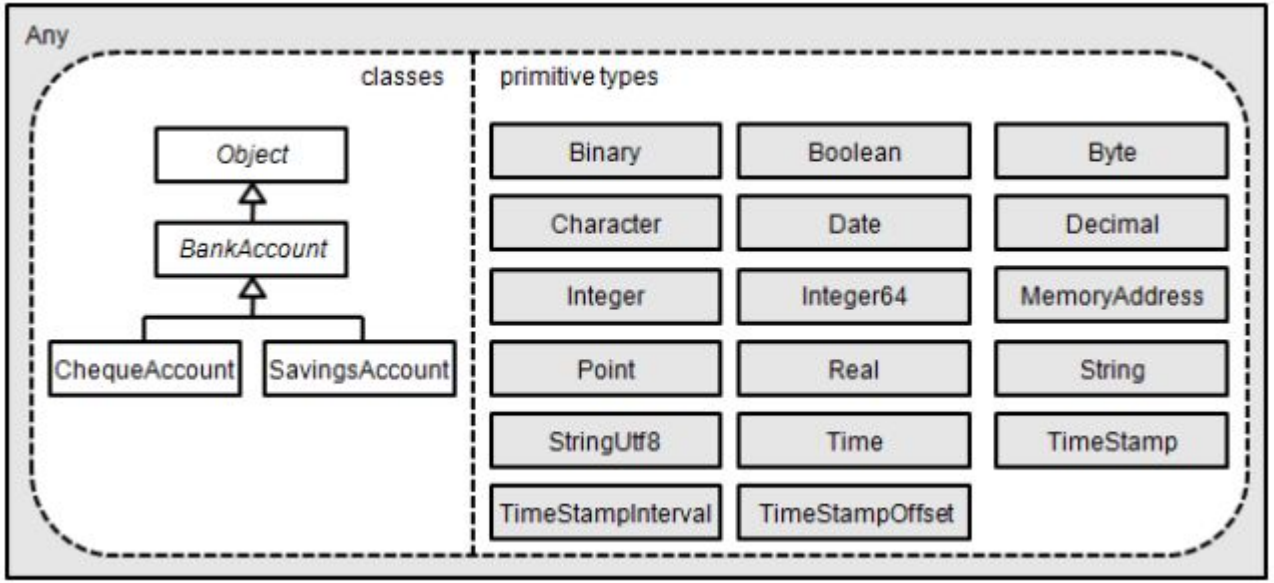
Hello world


Methods Browser for FirstScheme::JadeScript
All
Instance
Type
FirstScheme
Object
Application
Global
JadeScript
WebSession
Class Details
Property Details
Class: RootSchema::JadeScript (107)
Superclass: Object
Access: public
Type: real
Lifetime: transient shared-transient
Volatility: Volatile
Default: transient
Maps: _system
Jade Method Definition for JadeScript
Reimplement Superclass Method
Name
helloWorld
Updating
Abstract
Protected
Subschema Hidden
Type Method
Final Settings
Final
Subschema Final
Subschema Copy Final
Execution Location
Default
Server
Client
Enter Text...
OK
Next
Cancel
Help
(Created on 01 January 1970, 03:00:00)

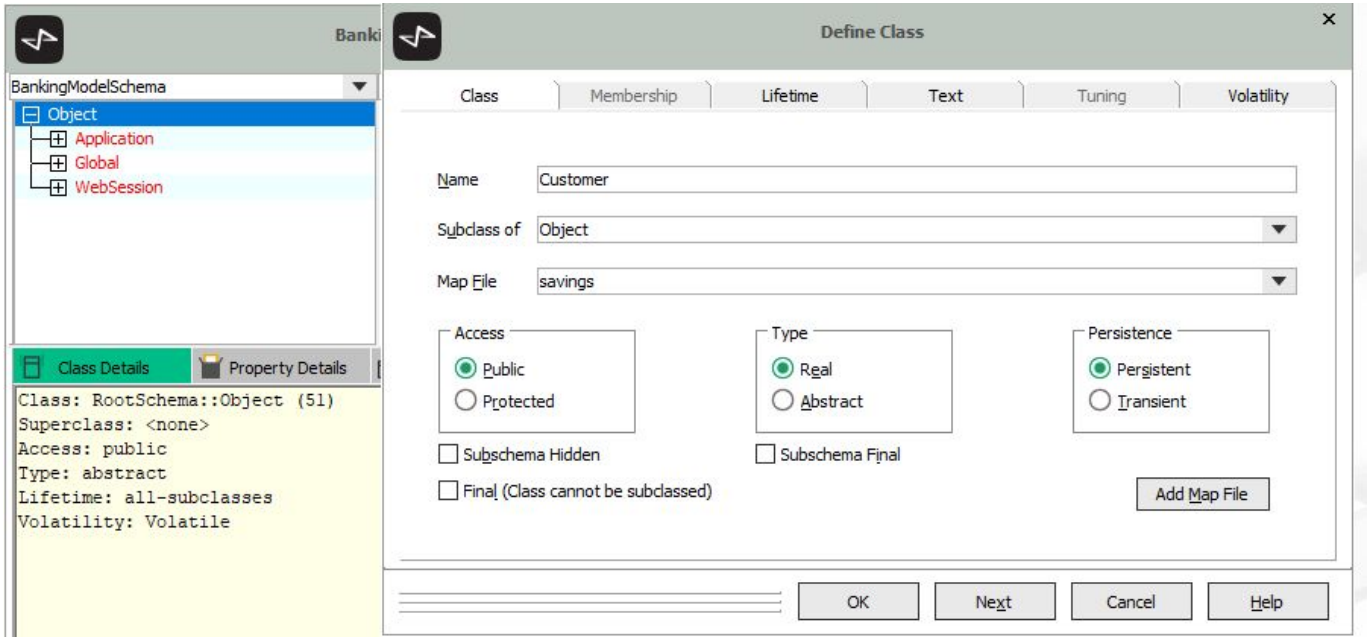
Hello world

```
1   helloWorld();
2
3   vars
4
5   begin
6       write "Hello World";
7   end;
8
```

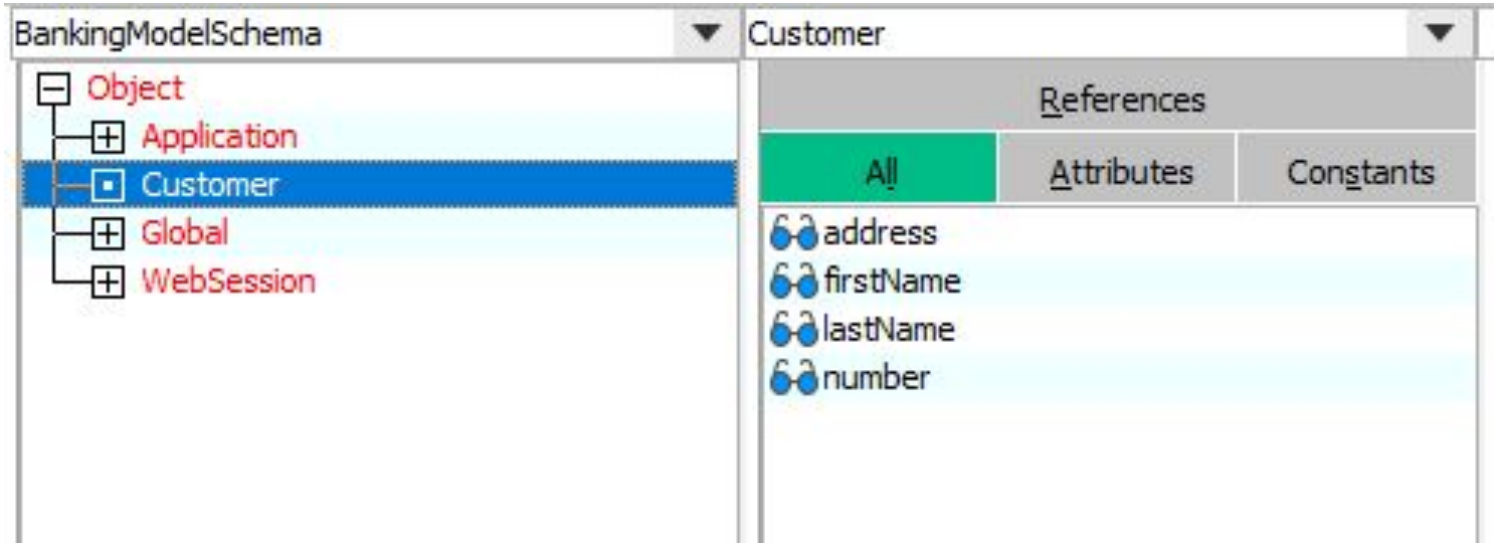
Типы


Any
classes
primitive types
Object
BankAccount
ChequeAccount
SavingsAccount
Binary
Boolean
Byte
Character
Date
Decimal
Integer
Integer64
MemoryAddress
Point
Real
String
StringUtf8
Time
TimeStamp
TimeStampInterval
TimeStampOffset

Создание класса



Создание класса. Атрибуты



Создание класса. Конструктор

```
1  create(addr, first, last: String) updating;
2  begin
3      self.address := addr.trimBlanks();
4      self.firstName := first.trimBlanks();
5      self.lastName := last.trimBlanks();
6  end;
7
```

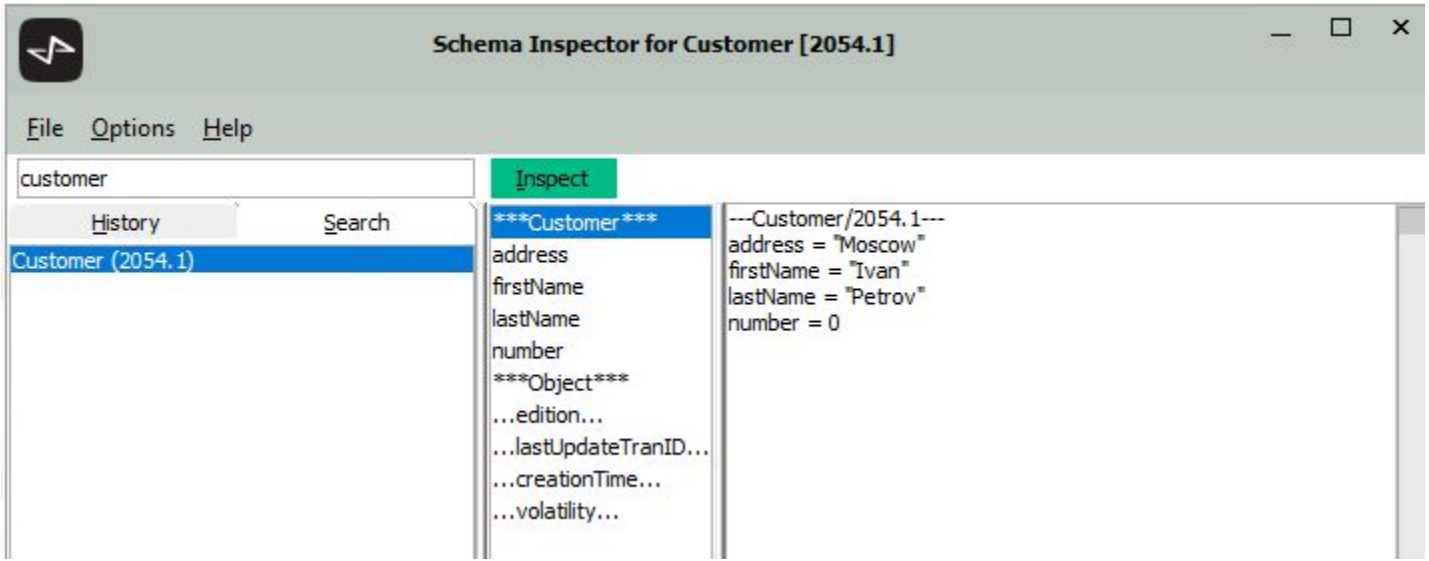
Добавление в базу

```
1  createCustomer();
2  vars
3  cust : Customer;
4  begin
5  beginTransaction;
6      cust := create Customer("Moscow", "Ivan", "Petrov") persistent;
7  commitTransaction;
8  end;
9
```

- **beginTransaction и commitTransaction** - обрамляют транзакции.

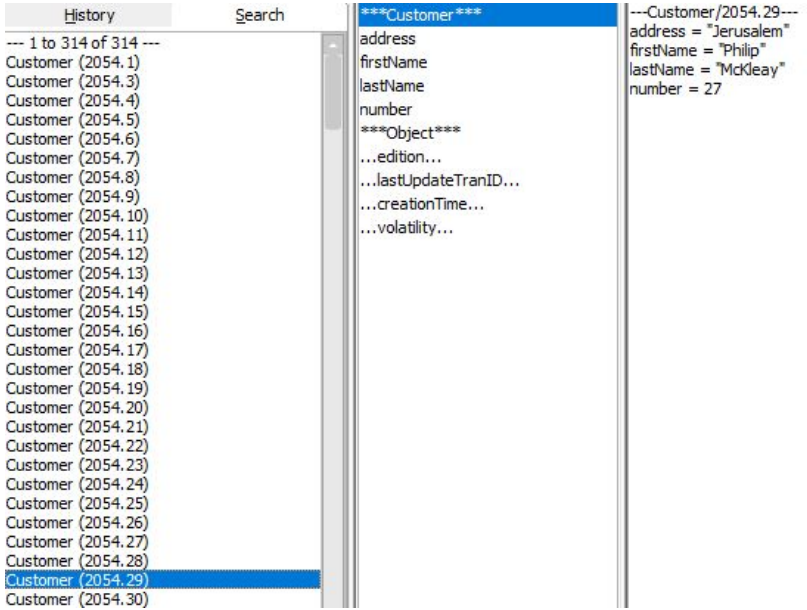
Добавление в базу

После добавления можно посмотреть что мы добавили



Загрузим из файла

```
1   loadCustomers();
2
3   vars
4       file: File;
5       str: String;
6       cust: Customer;
7
8   begin
9       app.init();
10      create file transient;
11      file.fileName := "D:\JadeCourse\Files\Customers.txt";
12      while not file.endOfFile() do
13          str := file.readLine();
14          beginTransaction;
15          cust := create Customer(str[41:end], str[16:25], str[1:15]);
16          commitTransaction;
17      endwhile;
18
19  end;
```



Пример с dictionary.

Загрузим в словарь данные

```
loadCustomers();

vars
    file: File;
    str: String;
    cust: Customer;
    dict: CustomersDictionary;
begin
    app.init();

    beginTransaction;
    create dict persistent;
    commitTransaction;

    create file transient;
    file.fileName := "D:\JadeCourse\Files\Customers.txt";
    while not file.endOfFile() do
        str := file.readLine();
        beginTransaction;
        cust := create Customer(str[41:end], str[16:25], str[1:15]);
        dict.add(cust)
        commitTransaction;
    endwhile;

end;
```

Пример с dictionary.

Выведем на экран фамилии всех покупателей из словаря.

```
printLastNames();

vars
    cust: Customer;
    coll: CustomersDictionary;
begin
    coll := CustomersDictionary.firstInstance();
    foreach cust in coll do
        write cust.lastName;
    endforeach;
end;
```

Продолжение изучения

- https://secure.jadeworld.com/JADETech/Education/InstallAdminCourse/JadeInstallAdminCourse.pdf - для администрирования.
- https://secure.jadeworld.com/JADETech/Education/DevCourse/JadeDevCourse.pdf - для разработчиков.
- https://github.com/jadesoftwarenz - github с примерами