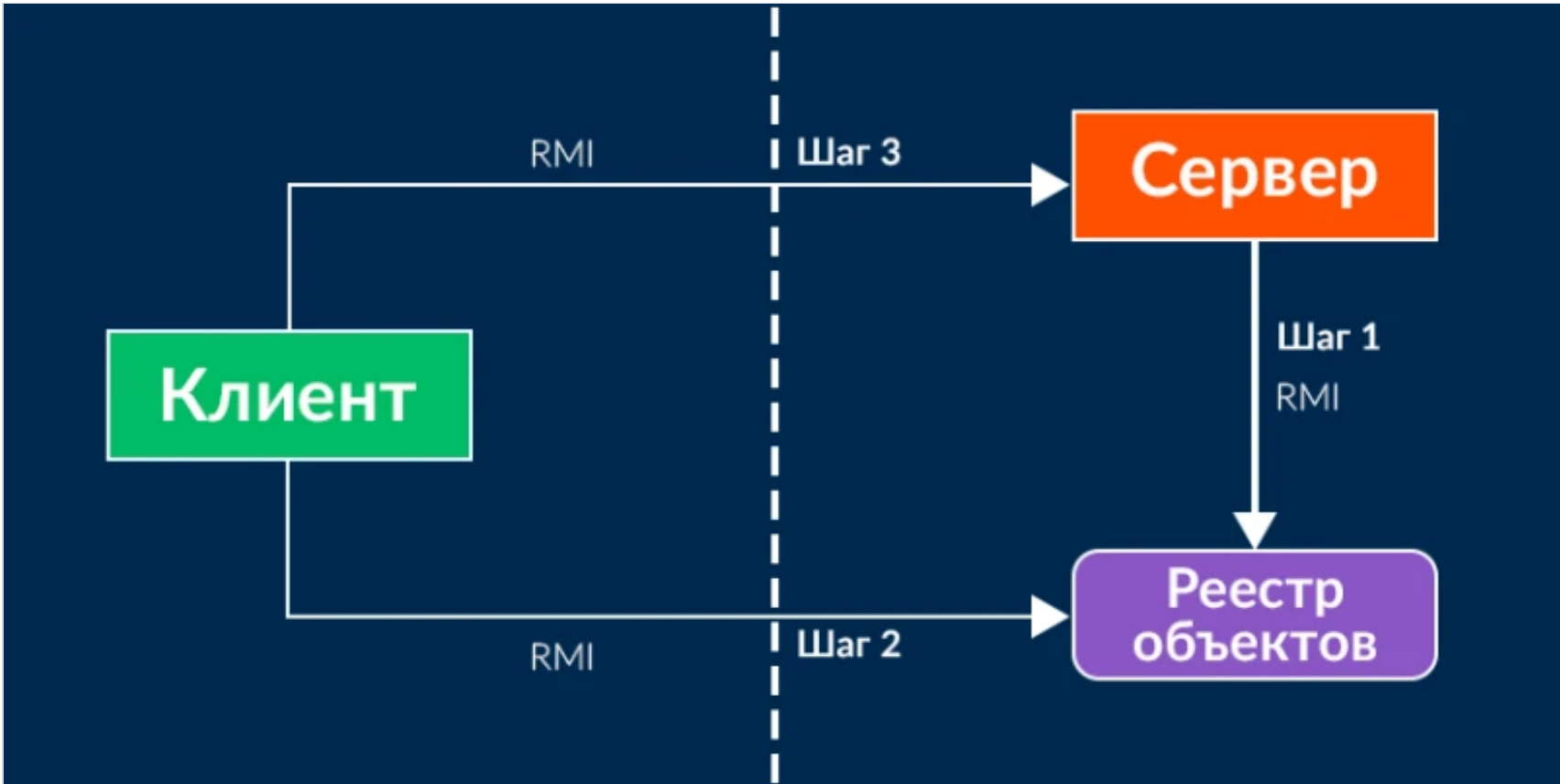


# RMI

Java Collections  
2 уровень, 9 лекция

ОТКРЫТА

— Привет! И еще одна радостная тема – **RMI**. **RMI** расшифровывается **Remote Method Invocation** – удаленный вызов методов. Или другими словами **RMI – это механизм, который позволяет объекту в одной Java-машине вызывать методы объекта в другой Java-машине**, даже если они находятся на разных компьютерах, в разных странах, на разных сторонах земного шара.



— Ничего себе! Звучит очень круто.

— Ага. Но я постараюсь дать только обзорную лекцию. Тут, если глубоко копать, можно запутаться в нюансах работы.

Но если не ударяться в крайности, то RMI не только очень прост, но и значительно упрощает жизнь программиста. За что ему глубокий респект.

Итак, мы хотим, чтобы один объект, находящийся в одной Java-программе, смог вызвать метод у объекта, находящегося в другой Java-программе. Где бы эти программы запущены ни были.

Мы рассмотрим самый простой пример, когда обе программы запущены на одном компьютере. **Чтобы программы могли взаимодействовать через интернет, необходимы настройки в правах Java-машины**, но сегодня мы это рассматривать не будем.

В Java удаленно можно вызывать только методы интерфейсов, но не классов.

Итак, у нас есть две программы, как же им вызывать методы друг друга?

Давай рассмотрим ситуацию, **когда одна программа содержит в себе некоторый объект, а вторая хочет вызвать его методы. Назовем первую программу – сервером, а вторую – клиентом.**

Я сначала дам пример кода, а потом мы его разберем.

— А что будет делать наша программа?

— Гм. Ну, давай для простоты, у программы будет один метод, который разворачивает переданную ему строку задом наперед.

— Вроде ничего.

Сначала нам понадобится интерфейс, который будет удовлетворять нашим требованиями:

Интерфейс для межпрограммного взаимодействия	
1	<code>interface Reverse extends Remote</code>
2	<code>{</code>
3	<code>    public String reverse(String str) throws RemoteException;</code>
4	<code>}</code>

Я создал интерфейс Reverse и добавил ему интерфейс-маркер Remote, а также исключение RemoteException. В процессе вызова метода могут происходить незапланированные сбои – тогда будет кидаться это исключение.

Затем нам нужно написать серверный класс, который бы реализовывал этот интерфейс:

Класс для сервера	
1	<code>class ReverseImpl implements Reverse</code>
2	<code>{</code>
3	<code>    public String reverse(String str) throws RemoteException</code>
4	<code>    {</code>
5	<code>        return new StringBuffer(str).reverse().toString();</code>
6	<code>    }</code>
7	<code>}</code>

— Вижу. В этом методе, мы разворачиваем строку задом наперед.

— Ага.

А теперь надо сделать этот объект доступным для вызова с другой программы. Вот как это делается:

Шаринг объекта	
1	<code>public static final String UNIC_BINDING_NAME = "server.reverse";</code>
2	
3	<code>public static void main(String[] args) throws Exception</code>
4	<code>{</code>
5	<code>    //создание объекта для удаленного доступа</code>
6	<code>    final ReverseImpl service = new ReverseImpl();</code>
7	
8	<code>    //создание реестра расшареных объектов</code>
9	<code>    final Registry registry = LocateRegistry.createRegistry(2099);</code>
10	<code>    //создание "заглушки" – приемника удаленных вызовов</code>
11	<code>    Remote stub = UnicastRemoteObject.exportObject(service, 0);</code>
12	<code>    //регистрация "заглушки" в реесте</code>
13	<code>    registry.bind(UNIC_BINDING_NAME, stub);</code>
14	
15	<code>    //усыпляем главный поток, иначе программа завершится</code>
16	<code>    Thread.sleep(Integer.MAX_VALUE);</code>
17	<code>}</code>

Рассказываю по строкам.

**Строка 1** – в переменной UNIC\_BINDING\_NAME храним придуманное нами уникальное имя нашего удаленного объекта (объекта, который доступен удаленно). Если программа шарит несколько объектов, у каждого должно быть свое уникальное имя. Уникальное имя нашего объекта — «server.reverse».

**Строка 9** — создаем специальный объект – реестр. В нем надо регистрировать объекты, которые мы шарим. Дальше ими занимается Java-машина. 2099 – это порт (уникальный номер, по которому другая программа может обратиться к нашему реестру объектов).

Т.е. чтобы обратиться к объекту, надо знать уникальный номер реестра объектов (порт), знать уникальное имя объекта и иметь такой же интерфейс, как и тот, который реализовывает удаленный объект.

— Ясно. Что-то вроде – позвонить по телефону (нужен номер) и попросить Соню (имя объекта)?

— Да. Теперь дальше.

**Строка 11** – создание «заглушки». Заглушка – это специальный объект, который принимает информацию об удаленном вызове, распаковывает ее, десериализует переданные параметры методов и вызывает нужный метод. Затем сериализует результат или исключение, если оно было, и отправляет все это назад вызывающему.

— Ясно. Почти. Ты сказал, что «десериализует параметры метода». Значит, типами аргументов удаленного метода могут быть только сериализуемые?

— Ага. А как же иначе ты будешь пересылать их по сети? Есть, правда, и исключения – так называемые объекты, которые передаются по ссылке, но сегодня мы о них говорить не будем.

Скажем так, пересылать несериализуемые объекты нельзя, но если очень хочется, то можно. Но это хлопотное дело, знаешь ли.

— Ок.

— Тогда дальше.

**Строка 13** – регистрируем в реестре заглушку нашего объекта под уникальным именем.

**Строка 16** – усыпляем главный поток. Все удалённые вызовы обрабатываются в отдельных нитях. Главное, чтобы программа в это время работала. Так что тут просто отправляем главную нить спать, и всё.

— Ок.

— Отлично, тогда пример клиента:

Работа с удаленным объектом

```
1 public static final String UNIC_BINDING_NAME = "server.reverse";
2
3 public static void main(String[] args) throws Exception
4 {
5     //создание реестра расшареных объетов
6     final Registry registry = LocateRegistry.getRegistry(2099);
7
8     //получаем объект (на самом деле это проху-объект)
9     Reverse service = (Reverse) registry.lookup(UNIC_BINDING_NAME);
10
11     //Вызываем удаленный метод
12     String result = service.reverse("Home sweet home.");
13 }
```

Объясняю код по строкам:

**Строка 1** – **уникальное имя** удаленного объекта. Должно быть одинаковым на клиенте и сервере.

**Строка 6** – получение ссылки на «**Реестр удаленных объектов**» по порту 2099, т.е. такому же, как и у реестра у серверного приложения.

Строка 12 – вызываем методы интерфейса так, как будто объект был создан в этой же программе. Никакой разницы.

— Круто! Это ж теперь можно писать распределенные приложения. Или игры типа морского боя для Android.

— Побойся бога, Амиго, операционная система Android была запрещена в 27 веке после третьей попытки захватить мир. У роботов к ней вообще доступа нет. Вас же потом от нее не оттянешь. Будете бегать и кричать «Убить всех человеков!».

— Гм. Ладно. Хотя надо будет у Диего еще спросить. Мало ли, может он что-нибудь интересное про нее расскажет.

— Вот и спроси. Ладно, давай до завтра.

— Пока, Риша, спасибо за интересную лекцию.

 +94 

Комментарии (106)

популярные

новые

старые

JavaCoder

Введите текст комментария

Фарид Гулиев

Уровень 41, Днепр, Украина

2 июля, 17:51

...

Можно было усыпить поток на бесконечность так:

1

Thread.sleep((long) Double.POSITIVE\_INFINITY);

А не через INTEGER.MAX\_VALUE

Ответить

 0 

Hidden #213


Уровень 13 (Forever&Ever)

3 марта, 14:17

...

Радостно

Ответить

 0 

Джу

Уровень 35

9 декабря 2021, 14:19

...

<https://habr.com/ru/post/74639/>

Ответить

 0 

PaiMei in J#


Grand Master в Eagles' Claw

18 октября 2021, 14:17

...

Оставлю ссылочку на видосик по теме, а то как то сплошной текст, суховато)  
[КЛИК-КЛИК](#)

Ответить

 +3 

Sulf8

Уровень 36, Самара, Россия

11 марта, 22:11

...

голос сверхпротивный

Ответить

 +2 

Лиза Воренувкина

Уровень 43, Кривой Рог, Ukraine

23 сентября 2021, 19:58

...

Ой как не понятно .

Ответить

 +5 

Алексей Харимов

Уровень 35, Харьков, Украина

7 сентября 2021, 10:12

...

Ответить

Cobalt River

Уровень 24, Москва

4 октября 2021, 09:51

...

Мне кажется вы не поняли о чем речь, и вводите в заблуждение.  
Аквивация это часть RMI. Она устарела из-за развития вебсервисов и т.п. но это не значит, что RMI весь устарел.

Вышла 17 Java.  
Удаление RMI Activation (JEP 407)

Механизм RMI Activation, который стал deprecated в Java 15, теперь удалён окончательно. RMI Activation был частью RMI, которая использовалась в чрезвычайно малом количестве приложений и фактически устарела. Для того чтобы избежать дорогой поддержки никому не нужного механизма, было принято решение его полностью удалить

Ответить

+3

Алексей Халимов

Уровень 35, Харьков, Украина

5 октября 2021, 22:56

...

так а что не так я сказал?)

Ответить

+3

Алексей

Уровень 23, Минск, Беларусь

24 мая 2021, 23:39

...

"Но это хлопотное дело" - все, кто из Беларуси, улыбнулись на этой фразе?)

Ответить

+35

Игорь Ходыко

Enterprise Java Developer

28 мая 2021, 23:35

...

Тут около лайка есть выбор какой этот лайк будет, так вот один из выборов- это "тянет на статью"))

Ответить

+18

Waumok

Backend Developer

24 августа 2021, 19:23

...

Я не из Белоруссии, но я программист)) потому загуглил и поржал))  
могу сказать только что —это было **не** "хлопотное дельце"

Ответить

+3

Anonymous #2489173

Уровень 35

26 марта 2021, 05:04

...

А вообще не особо понимаю первую схему.  
То есть вызов метода начинается не с запроса клиента к серверу а с взаимодействия между Сервером и реестром объектов? А откуда сервер узнаёт, что надо взаимодействовать, если запроса ещё не было?

Ответить

0

Ars

Уровень 41

17 ноября 2021, 18:54

...

Ниоткуда.  
1. Сначала сервер в реестре регистрирует объекты, которые можно вызывать удалённо.  
2. Потом клиент получает из реестра ссылку на объект  
3. И вызывает объект.

Ответить

+1

Anonymous #2489173

Уровень 35

13 декабря 2021, 06:03

...

а реестр не на сервере лежит? откуда клиент получает ссылку?  
если не на сервере и оно всё локально, то откуда в реестре появляются данные об этих методах?

Ответить

0

Ars

Уровень 41

13 декабря 2021, 12:50

...

В данном случае реестр лежит на сервере. Шаг 1 можно рассматривать как подготовительный. Сервер сначала создаёт реестр, заносит туда объекты и ждёт пока что-нибудь их вызовет. Непосредственно взаимодействие начинается когда клиент обращается к реестру для получения ссылки на объект.

Ответить

0

Anonymous #2489173

Уровень 35

26 марта 2021, 05:02

...

В последнее время сразу иду в комментарии, потому что там обычно больше полезной информации, чем в лекциях

Ответить

+3

Pig Man

Главная свинья в Свинарнике

5 февраля 2021, 21:59

...

1. Запускаем предоставленный в лекции код, получаем:

1

ExportException: Port already in use

```
1 // Меняем это:
2 Registry registry = LocateRegistry.createRegistry(2099);
3 // На это:
4 Registry registry = LocateRegistry.getRegistry(2099);
```

3. Удивляемся, какого хрена там вообще был прописан метод createRegistry()

Ответить +11

**Dmitry Vasilyev** Уровень 26, Саратов, Россия 17 апреля 2021, 10:11

Товарищи из администрации - обратите внимание. Либо на вас клеветают, либо у вас в статье с незапамятных времён не исправлена неточность.

Ответить 0

**Pig Man** Главная свинья в **Свинарнике** 17 апреля 2021, 11:13

Оставь их, они квесты по sql и jsp усердно делают. Отвлечешь их и это еще на 10 лет затянется

Ответить +10

**Dmitry Vasilyev** Уровень 26, Саратов, Россия 17 апреля 2021, 11:48

Аргумент.

Ответить 0

**Pavlo Plynko** Java-разработчик в **JavaRush** **EXPERT** 22 апреля 2021, 18:29

Поправили, спасибо)

Ответить +3

Показать еще комментарии

ОБУЧЕНИЕ

- Курсы программирования
- Курс Java
- Помощь по задачам
- Подписки
- Задачи-игры

СООБЩЕСТВО

- Пользователи
- Статьи
- Форум
- Чат
- Истории успеха
- Активности

КОМПАНИЯ

- О нас
- Контакты
- Отзывы
- FAQ
- Поддержка



JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский

СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ



