

# Средства Python для создания Jabber сервисов и транспортов

Добров Сергей

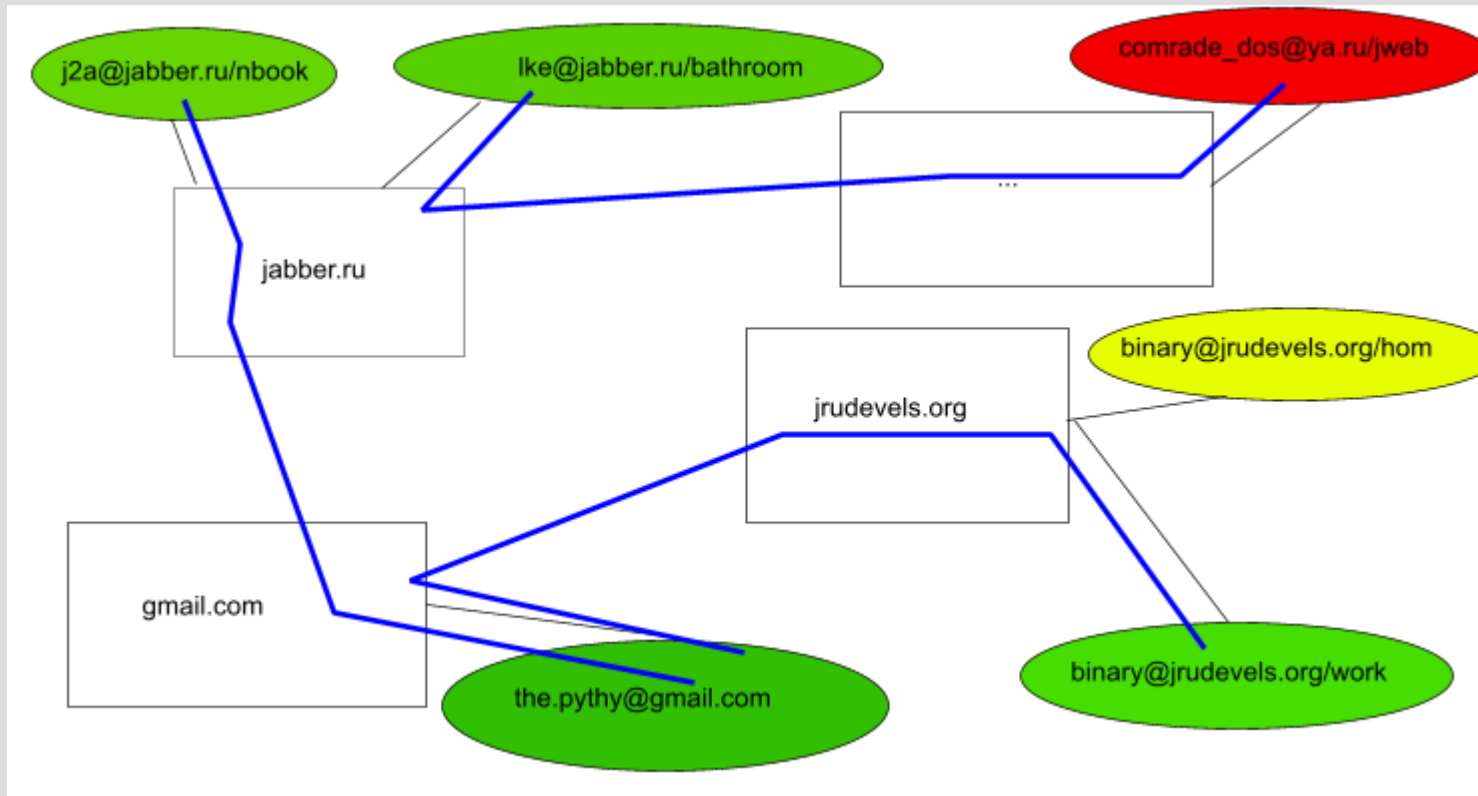
<http://jrudevels.org/>

RuPyRu 2008

# 1. Что такое Jabber/XMPP?

- XMPP  
(eXtensible Message and Presence Protocol)  
протокол:
  - Открытый
  - Стандарт (RFC 3920-3923)
  - Ядро – RFC
  - Расширения - XEP
- Jabber – IM-платформа на XMPP

# 1.1. Структура Jabber-сети.

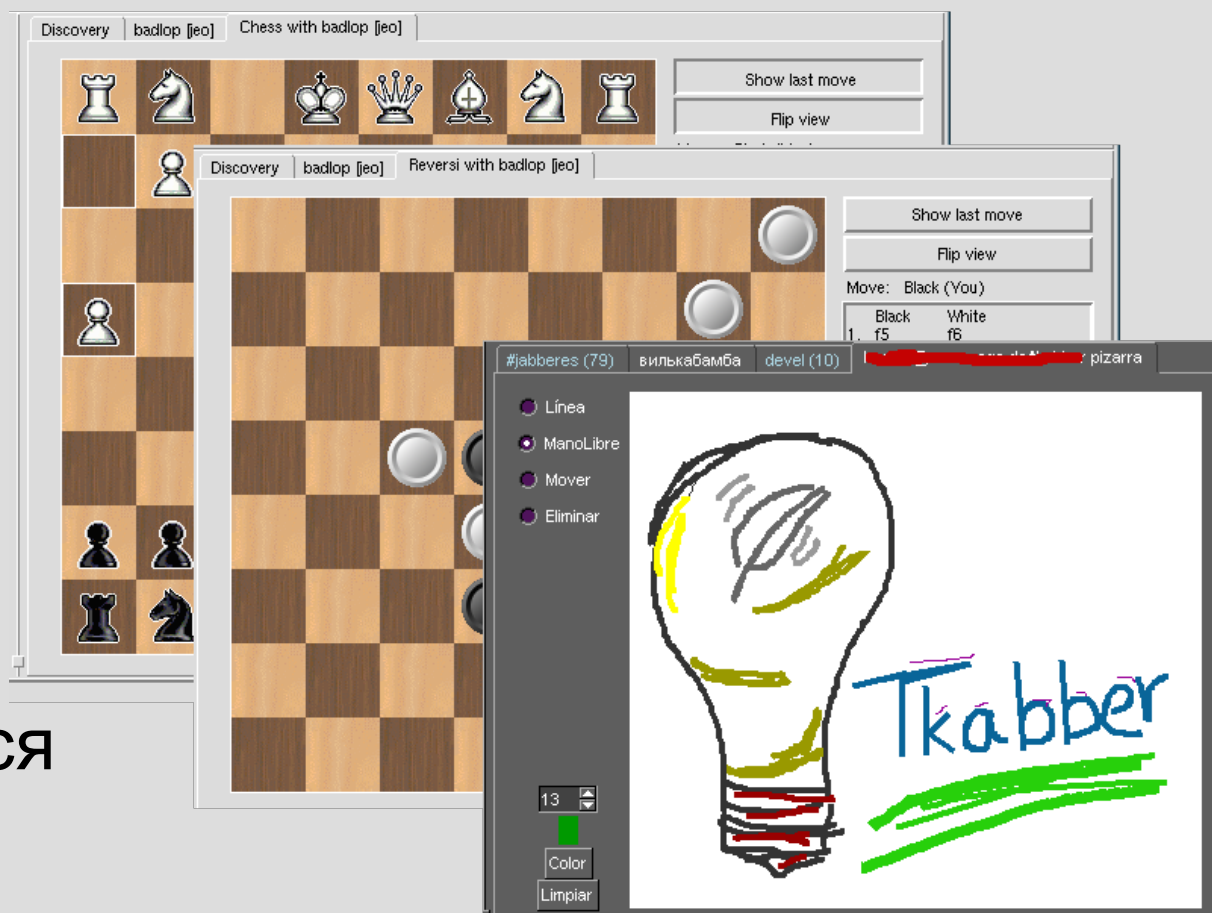


## 1.2. Jabber умеет

- Шифрование всего и вся
  - Соединение от клиента к серверу
  - Межсерверное соединение
  - GPG/GnuPG каждого сообщения (клиент)
- Подпись
  - Статус (клиент)
  - Сообщения (клиент)
- Списки приватности;
- Групповые чаты;

# 1.3 Jabber может

- Игры
  - Шашки
  - Шахматы
  - Реверси
  - Сапёр
- Совместно
  - Рисовать
  - Общаться
  - Обмениваться файлами



# 1.4. Основные виды информационных пакетов.

- XMPP – XML
- Пакеты aka стансы (stanza):
  - <message> - сообщения
  - <presence> - информация о статусе/подписке
  - <iq> - пинг-понг (ответ будет всегда)

# 1.4.1. Примеры информационных пакетов.

- ```
<message  
  to='j2a@jabber.ru'  
  from='comrade_dos@ya.ru'  
  type='chat'>  
<body>Привет!</body>  
</message>
```
- ```
<presence  
  to='j2a@jabber.ru'  
  from='comrade_dos@ya.ru'  
  type='unavailable' />
```

[20:18] Comrade DOS: Привет!

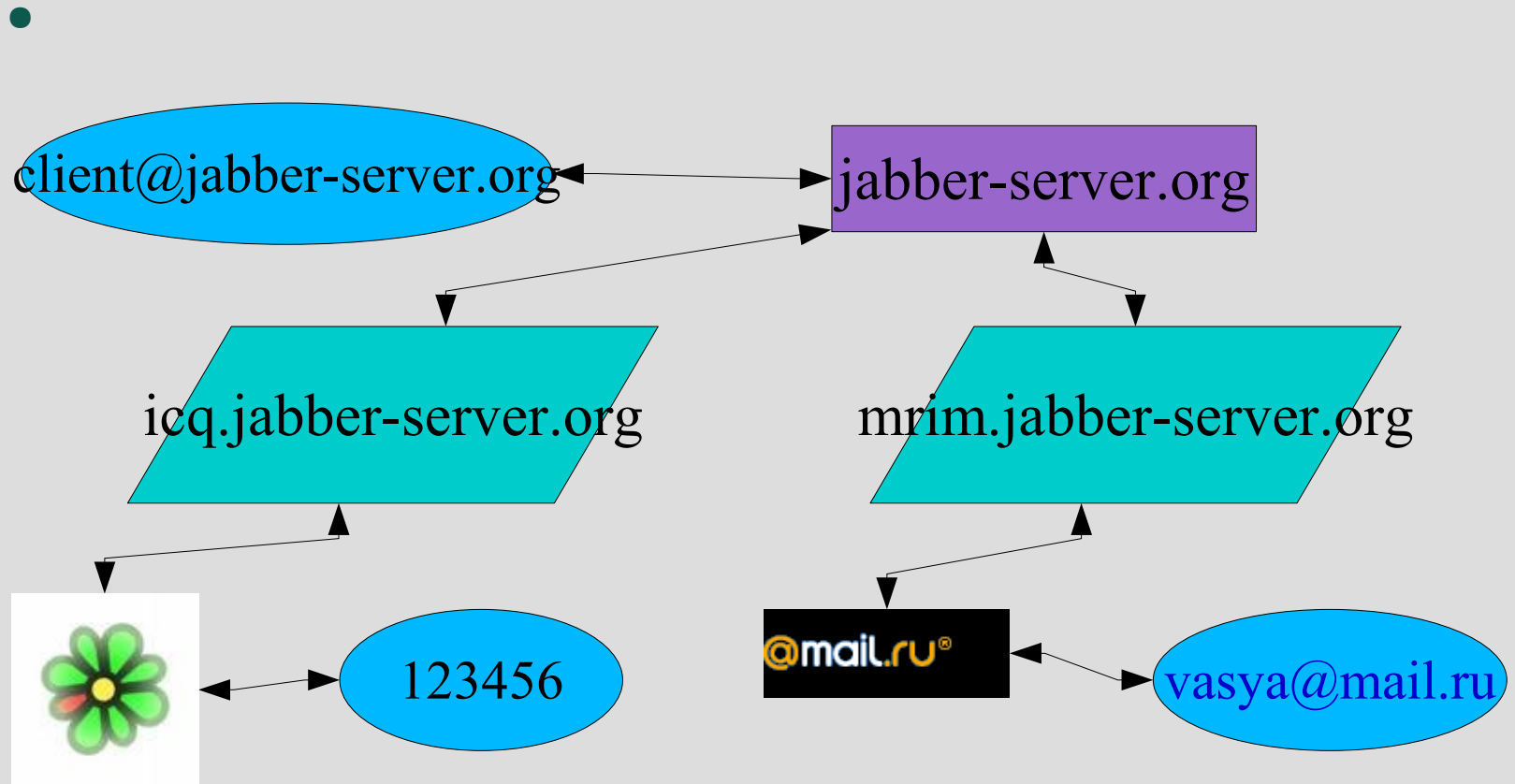


# 1.5. Транспорты и сервисы.

- Транспорты:
- ICQ;
- Yahoo;
- MRIM;
- E-Mail;
- И даже Jabber/XMPP.
- Сервисы:
- Переводчики;
- Словари;
- Хранилища файлов;
- Рассылки;
- другое.



# 1.5.1. Работа транспортов.



## 2. Основные средства разработки Jabber-сервисов.

- Т.к. Протокол основан на XML, мы так или иначе сталкиваемся с XML-парсерами и средствами работы с XML. Есть несколько способов написать Jabber-сервис на Python:
- Использовать любой доступный XML-парсер и вручную контролировать соединение с Jabber-сервером;
- Библиотека рухтпрр;
- Библиотека хтпрру;
- Библиотека Twisted.

## 2.1. Использование обычного XML-парсера.

- Написание сервиса или транспорта не подразумевает использования шифрования канала до сервера и использования сложных схем аутентификации, поэтому для написания небольших сервисов вполне можно обойтись и обычным XML-парсером.

## 2.2. Использование RuXMPP.

- Удобная библиотека для программирования XMPP-сервисов;
- Предоставляет классы для абстракции от очень многих понятий XMPP;
- Использует libxml2;
- Очень подходит для быстрого написания небольших сервисов.

## 2.2.1. Плюсы и минусы.

- Плюсы:
- Простая в обращении библиотека, очень хорошая документация;
- Подходит для быстрого написания сервисов.
- Минусы:
- Использование libxml2, который имеет утечки памяти;
- Проблемы с блокировкой обработки соединения при длительной обработке запроса пользователя.

## 2.2.2. Использование.

- ```
def message(self, stanza):  
    If stanza.get_type=='error': return True  
    m=Message(to_jid=stanza.get_from(),  
from_jid=stanza.get_to(), stanza_type=stanza.get_type(), body=u'hi!')  
    return True
```
- Переводчик на базе [translate.ru](http://translate.ru) - [translate.jrudevels.org](http://translate.jrudevels.org);
- RSS-ридер [rss.jrudevels.org](http://rss.jrudevels.org);
- [Habahaba.jrudevels.org](http://Habahaba.jrudevels.org).

## 2.3. Использование хтррру.

- Библиотека, предоставляющая классы для абстракции от понятий XML, построенная на собственном парсере `simplexml.py`

## 2.3.1. Плюсы и минусы.

- Плюсы:
- Удобная работа с XML через объектно-ориентированную модель, например:  
`message.T.message.T.body.setData('hi!')`
- Хорошая документация.
- Минусы:
- Проблемы с блокировкой обработки соединения при длительной обработке запроса пользователя.

## 2.3.2. Пример ИСПОЛЬЗОВАНИЯ.

- ```
def message_cb(self, xmpp_session, message):  
    if message.getType() not in ['chat', None]:  
return  
    m=Message('target@jid.com', 'Hello!')  
    xmpp_session.send(m)  
    raise xmpp.NodeProcesse
```
- Транспорт в Mail-Ru Агент.

## 2.4. Использование Twisted.

- Twisted – асинхронная платформа для построения сетевых приложений. Поддерживает XMPP в составе модуля Twisted-Words. Использует собственный парсер Xish на основе модели DOM и XPath.

## 2.4.1. Плюсы и минусы.

- Плюсы:
- Асинхронная модель позволяет избавиться от проблем блокировок при длительной обработке запроса;
- Хорошая стабильность.
- Минусы:
- Не очень полная документация;
- Плохое реагирование разработчиков на баг-репорты;
- Сравнительно малый уровень абстракции.

## 2.4.2. Использование.

- ```
def onMessage(self,el):  
    if el.getAttribute('type') in ['chat',None]:  
        msg=Element((None,"message"))  
        msg.Attributes['to']='to@server.org'  
        msg.Attributes['type']='chat'  
        msg.addElement("body",content='hi')  
        self.send(msg)
```
- Транспорт J2J ([j2j.jrudevels.org](http://j2j.jrudevels.org));
- Транспорт PyICQ-t.

## 3. Заключение.

- Python является удобным средством для быстрого и эффективного программирования IM-сервисов и транспортов для Jabber. Однако, проработка библиотек для этого пока еще оставляет желать лучшего. Основываясь на задачах и сроках, нужно выбрать хороший компромисс между качеством и скоростью разработки.

## 4. Контакты.

- Если Вас заинтересовала технология XMPP/Jabber, Вы можете узнать большего на нашем ресурсе разработчиков <http://JRuDevels.org>:
- <http://forum.jrudevels.org> - форум разработчиков и пользователей;
- <http://jawiki.jrudevels.org> - русскоязычная WiKi по Jabber/XMPP;
- <xmpp:jrd@conference.jabber.ru> - онлайн конференция JRuDevels.