Приложен програмен интерфейс JavaMail (JavaMail API)

API интерфейс JavaMail

- Осигурява платформено и протоколно независим фреймуърк за изграждане на приложения за поща и съобщения.
- Осигурява набор от абстрактни класове и подкласове, които са необходими за разработката една пощенска система.

http://www.ietf.org/rfc.html

- Simple Mail Transfer Protocol (SMTP) - RFC 821
- Post Office Protocol - RFC 1939
- Internet Message Access Protocol - RFC 2060
- Multipurpose Internet Mail Extensions – RFC 1521, RFC 1522

SMTP

- Определя механизма за доставка на електронна поща.
- JavaMail базирани програми комуникира с доставчик на Интернет услуги (ISP's) на SMTP сървър.

POP

- Механизъм за получаване на електронна поща.
- Дефинира поддръжката на пощенска кутия за всеки потребител.

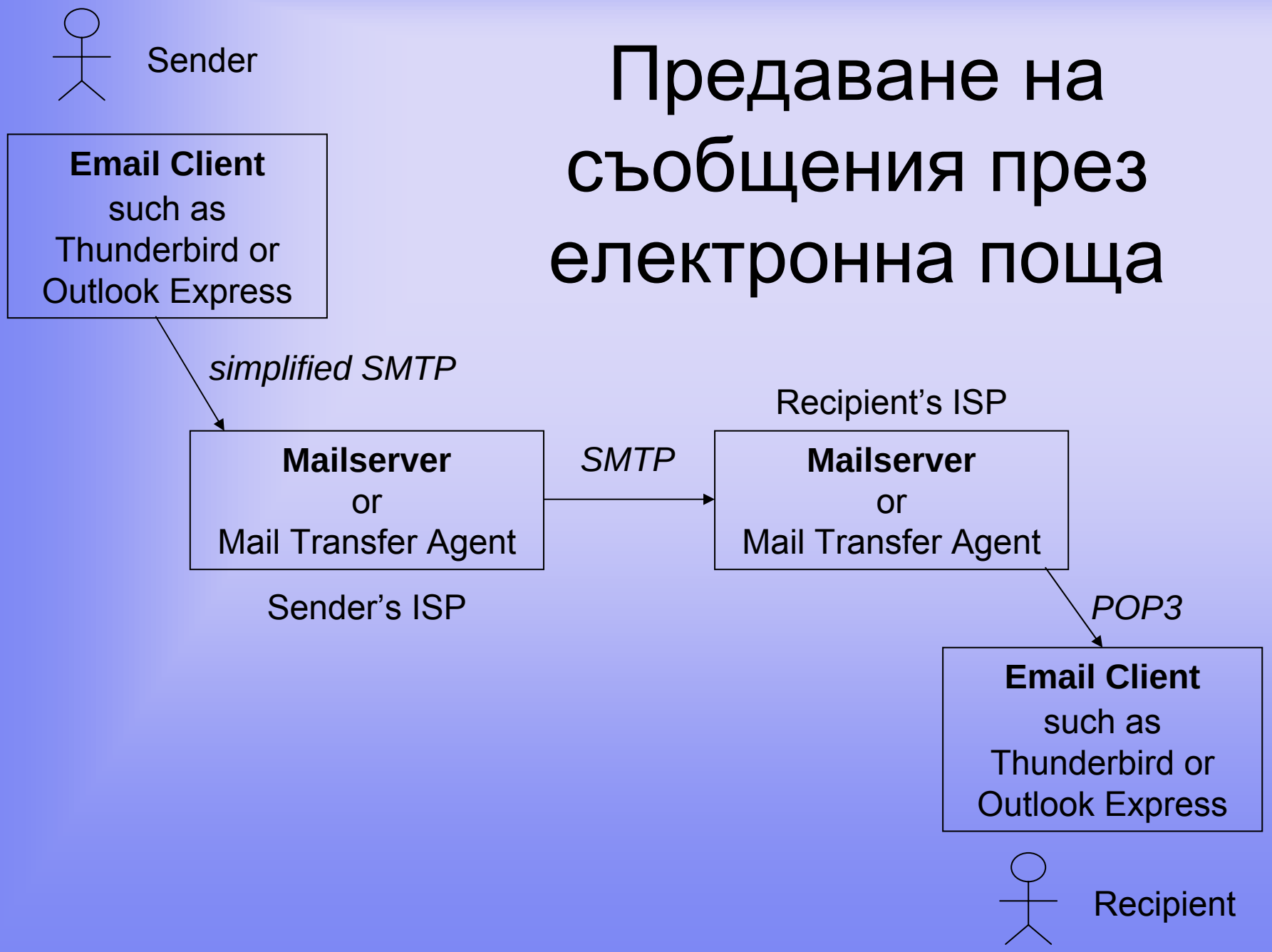
IMAP

- IMAP е по-комплексен протокол за получаване на съобщения.
- При използването на IMAP протокола се сървъра за електронна поща трябва да го поддържа.
- Той е с повече функционалност за мейл сървър, като изисква сървъра да получава на нови съобщения, дава ги на потребители, когато е необходимо, и ги поддържа в множество папки за всеки потребител.

MIME

- Не е протокол за обмяна.
- Дефинира съдържанието на съобщението което се обменя.

Предаване на съобщения през електронна поща



Основни класове javax.mail

- Session
- Message
- Address
- Authenticator
- Transport
- Store
- Folder

Изпращане на съобщение

1. Задаване на свойството mail.host да сочи към локалния пощенски сървър.
2. Започване на сесия чрез метода: Session.getInstance().
3. Създаване на нов обект за съобщение.
4. Задаване на адреса на подателя.
5. Задаване на адреса на получателя.
6. Задаване на темата.
7. Задаване на съдържанието на съобщението.
8. Изпращане на съобщението чрез метода: Transport.send( ).

Пример

```
Properties props = new Properties( );
props.put("mail.host", "smtp.fmi.uni-
  sofia.bg");
Session mailConnection =
  Session.getInstance(props,
     null);

Message msg = new
  MimeMessage(mailConnection);

Address sender = new
     InternetAddress("adelina@abv.bg",
                     "Adelina");
```

```
Address receiver =  new
  InternetAddress(“aaa@gmail.com");

msg.setSubject("Hello");
msg.setContent("Hello again!",
                              "text/plain");
msg.setFrom(sender);
msg.setRecipient(Message.RecipientType.TO,
                                  receiver);

Transport.send(msg);
s
```

Получаване на съобщение

1. Задаване на стойности на връзката.
2. Построяване на Authenticator.
3. Вземете Session обекта с метода getDefaultInstance().
4. Използвайте метода getStore( ), за да върне обекта от класа Store.
5. Връзвате се към услугата за съхранение.
6. Взимате INBOX директорията за съхраняване чрез извикване на метода getFolder().
7. Отваряне на директорията INBOX.
8. Извличане на съобщенията от директорията, като масив от Message обекти.
9. Обхождане на масива със съобщения и ги обработвате чрез методите на класа Message.
10. Затваряне на folder и store.

```
import java.util.*;

public class POP3Client {

  public static void main(String[] args) {

    Properties props = new Properties( );

    String host = "mail.fmi.uni-sofia.bg";

    String username = "adelina";
    String password = "123";
    String provider = "pop3";

    try {
     // Connect to the POP3 server
      Session session = Session.getDefaultInstance(props, null);
      Store store = session.getStore(provider);
      store.connect(host, username, password);

      // Open the folder
      Folder inbox = store.getFolder("INBOX");
      if (inbox == null) {
        System.out.println("No INBOX");
        System.exit(1);
      }

      inbox.open(Folder.READ_ONLY);
```

Пример

```
    // Get the messages from the server
    Message[] messages = inbox.getMessages( );
    for (int i = 0; i < messages.length; i++) {
      System.out.println("------------ Message " + (i+1) + " ------------");
      messages[i].writeTo(System.out);
    }

    // Close the connection but don't remove the messages from the server
    inbox.close(false);
    store.close( );
}       catch (Exception ex) {
    ex.printStackTrace( );

}
```