|  |
| --- |
| Dept. of CSE, Pusan National University |
| UTF-8을 사용한 다국어 ID를 지원하는 메일서버 구현 최종보고서 |
| 소프트웨어시스템(김 경 석 교수님) |
|  |
| 권 상혁(200624410) 송 영찬(201024575) 염 재걸(201024576) 윤 정용(201024578) Team Name : 네 얼간이(4 idiots) |
| **2013-09-27** |

|  |
| --- |
| 이 문서는 2013년도 졸업과제 주제인 UTF-8을 사용하여 다국어 ID를 지원하는 메일서버를 구현에 관하여 중간기간까지의 설계상황 및 상세 내용을 담고 있습니다. |

UTF-8을 사용한 다국어 ID를 지원하는 메일서버 구현 최종보고서

**팀명 :네얼간이**

**권 상혁(200624410) 송 영찬(201024575) 염 재걸(201024576) 윤 정용(201024578) Team Name : 네 얼간이(4 idiots)**

Dept. of CSE, Pusan National University

2013년 10월 28일

**요 약**

이 문서는 2013년도 졸업과제 주제인 UTF-8을 사용하여 다국어 ID를 지원하는 메일서버를 구현에 관하여 중간기간까지의 설계상황 및 상세 내용을 담고 있습니다.

주제어:UTF-8, Mail Server, ASCII, RFC 규격, SMTP, James …

목차

1. 요구조건 및 제약 사항 분석에 대한 수정사항--------------------2
2. 요구조건
3. 제약사항
4. 수정사항
5. 과제의 목표---------------------------------------------2
6. 관련 기술 정의------------------------------------------3
7. 설계 상세화 및 변경 내역 ----------------------------------3
8. 변경 내역
9. 설계 상세화
10. 수행내용 및 실험결과-------------------------------------13
11. local 환경에서 메일 송수신
12. 외부서버와의 메일 송수신
13. 최종결과
14. 추진 체계----------------------------------------------20
15. 추진 일정----------------------------------------------21
16. **요구조건및제약 사항 분석에 대한 수정사항**
17. 요구조건

* 메일 전송 및 수신을 위한 메일서버 환경 구현
* 사용자의 계정 및 메일서버를 DB와 연동하여 관리할 수 있게 구현
* 메일을 보내고 받을 수 있는 WebMail Page를 구현
* 사용자의 ID를 UTF-8을 사용하여 한글 및 다국어지원이 가능하도록 구현

1. 제약사항
   * + UTF–8을 사용해서 다국어 지원이 가능해야 한다.
     + Domain파트는 따로 encoding을 할 필요 없음
     + 메일서버를 OpenSource인 James를 이용하여 구현
     + 개인의 PC를 메일서버로 사용하게 되고, PC간 메일 전송이 가능해야 한다
     + 외부 메일서버인 Gmail, Naver, Daum 등과
2. 수정사항

* 착수보고서에서 제시되었던 Downgrade 및 개발환경에 대한 내용을 수정하였고, 여러 번의 시행착오 끝에 Windows 환경에서 Java 및 JSP를 사용하여 해당 주제의 구현을 진행하였다.
* 초기에 계획하였던 Android application을 구현하는 것 보다, WebMail 페이지를 구현하여 시간을 들이는 것이 더욱 완성도를 높일 수 있겠다는 판단하에 Android관련 내용은 삭제하였다.

1. **과제의 목표**

현재 일반적으로 사용되고 있는 메일시스템에서는 메일 주소의 local 파 트를 아스키형식의 문자만 지원하고 있다

RFC 6530 문서에 의하면

In order to use internationalized email addresses, it is necessary to

internationalize both the domain part and the local part of email

addresses. The domain part of email addresses is already

internationalized [RFC5890], while the local part is not. Without

the extensions specified in this document, the mailbox name is

restricted to a subset of 7-bit ASCII [RFC5321].

메일 주소의 도메인 파트는 이미 RFC5890에 의해 국제화 되어 있으며

반면에 로컬파트는 그렇지 못하여서 메일박스 이름은 7비트 아스키의 서브셋으로

제한이 되게 된다

본 과제의 목표는 이 local 파트를 UTF8을 사용하여 다국어로 사용할 수 있는 메일 시스템을 구축하는 것이다.

1. **관련 기술 정의**

**가)SMTP**

[인터넷](http://ko.wikipedia.org/wiki/%EC%9D%B8%ED%84%B0%EB%84%B7)에서 [이메일](http://ko.wikipedia.org/wiki/%EC%9D%B4%EB%A9%94%EC%9D%BC)을 보내기 위해 이용되는 프로토콜이다. 사용하는 TCP 포트번호는 25번이다. 상대 서버를 지시하기 위해서 [DNS](http://ko.wikipedia.org/wiki/%EB%8F%84%EB%A9%94%EC%9D%B8_%EB%84%A4%EC%9E%84_%EC%8B%9C%EC%8A%A4%ED%85%9C)의 MX레코드가 사용된다. [RFC](http://ko.wikipedia.org/wiki/RFC)2821에 따라 규정되어 있다. 메일 서버간의 송수신뿐만 아니라, 메일 클라이언트에서 메일 서버로 메일을 보낼 때에도 사용되는 경우가 많다. SMTP는 텍스트 기반의 프로토콜로서 요구/응답 메시지뿐 아니라 모든 문자가 7bit [ASCII](http://ko.wikipedia.org/wiki/ASCII)로 되어있어야 한다고 규정되어 있다. 이 때문에 문자 표현에 8비트 이상의 코드를 사용하는 언어나 첨부파일과 자주 사용되는 각종 바이너리는 마임(MIME)이라고 불리는 방식으로 7비트로 변환되어 전달된다. SMTP는 메세지를 생성하는 방법을 규정하지 않는다. 메세지 생성을 위하여 로컬 편집이나 단순한 전자 우편 [응용](http://ko.wikipedia.org/wiki/%EC%9D%91%EC%9A%A9)이 사용된다. 메세지가 생성되면 호출된 SMTP가 메세지를 받고 [TCP](http://ko.wikipedia.org/wiki/TCP)를 이용하여 다른 [호스트](http://ko.wikipedia.org/wiki/%ED%98%B8%EC%8A%A4%ED%8A%B8)의 SMTP에게 전달한다.

|  |
| --- |
| S: 220 www.example.com ESMTP Postfix  C: HELO mydomain.com  S: 250 Hello mydomain.com  C: MAIL FROM: <sender@mydomain.com>  S: 250 Ok  C: RCPT TO: <friend@example.com>  S: 250 Ok  C: DATA  S: 354 End data with <CR><LF>.<CR><LF>  C: Subject: test message  C: From: sender@mydomain.com  C: To: friend@example.com  C:  C: Hello,  C: This is a test.  C: Goodbye.  C: .  S: 250 Ok: queued as 12345  C: quit  S: 221 Bye  SMTP 통신의 예 |

**나)POP3**

POP3 (Post Office Protocol version 3)는 응용 계층 [인터넷](http://ko.wikipedia.org/wiki/%EC%9D%B8%ED%84%B0%EB%84%B7) 프로토콜 중 하나로, 원격 서버로부터 [TCP/IP](http://ko.wikipedia.org/wiki/TCP/IP) 연결을 통해 [이메일](http://ko.wikipedia.org/wiki/%EC%9D%B4%EB%A9%94%EC%9D%BC)을 가져오는데 사용된다. 여러 버전의 POP 프로토콜이 개발 되었지만 일반적으로 POP를 지칭하는 경우 보통 POP3을 가리킨다.[윈도 라이브 핫메일](http://ko.wikipedia.org/wiki/%EC%9C%88%EB%8F%84_%EB%9D%BC%EC%9D%B4%EB%B8%8C_%ED%95%AB%EB%A9%94%EC%9D%BC), [G메일](http://ko.wikipedia.org/wiki/G%EB%A9%94%EC%9D%BC), 및 [Yahoo](http://ko.wikipedia.org/wiki/Yahoo)! 메일과 같은 대부분의 웹 메일에서 지원한다.

대부분의 이메일 프로그램이 서버에 이메일을 남겨두는 기능을 제공하지만, POP는 원격 서버에 접속해서 이메일을 가져온 후 서버에서 이메일을 삭제한다. 반면 [IMAP](http://ko.wikipedia.org/wiki/IMAP)와 같은 다른 프로토콜은 이메일을 남겨두는 등 좀 더 복잡하고 많은 기능을 제공한다.

1. **설계 상세화 및 변경 내역**
2. 변경 내역

* 개발환경 변경내역
* 운영체제 : Windows 7 & Windows XP
* 개발언어 : Java, JSP
* 구현환경 : Eclipse
* 메일서버 : James 2.3.2
* 컴파일러 : Java Compiler
* Database : MySql 5.6
* Linux 운영체제를 사용해서 구현을 하려다 Windows 환경으로 변경
* 구현 language 변경 ( Python -> Java)
* Sendmail 구현 과정에서 OpenSource인 James 2.3.2 메일서버 사용으로 변경

1. 상세 설계내역

**James Mail Server**

|  |
| --- |
| SMTP(send)  SMTP(send)  **내 ISP’s**  **메일서버**  Internet  **내 컴퓨터**  **(Client)**  **Internet**  POP3(receive)  **메일서버**  POP3(receive)  그림 1. |

ISP’s 메일서버 부분이 SMTP프로토콜을 해석 할 수 있는 중간서버 역할을 하게 되는데, 이에 해당하는 서버가 James로 사용가능 하다.

James를 사용하여 자신의 PC를 메일서버로 구축하기 위해서는 우선적으로 Java가 설치되어 있어야 사용이 가능하였고, James 서버의 환경설정은 config.xml 파일을 통해서 제어 할 수 있었기 때문에 이 문서의 수정이 필요하다.

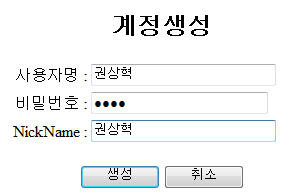
|  |
| --- |
| 그림 2 config.xml 파일에 IP주소 추가 |

위와 같이 파일을 수정 한 뒤에 James를 구동시켜보면 그림3과 같은 화면이 출력이 된다면 정상적으로 메일서버가 동작한다는 의미이다.

|  |
| --- |
| 그림3 메일서버 정상동작 화면 |
| **MySql연동**  앞서 구현했던 James를 MySql과 연동하여 DB에서 사용자관리 및 메일파일을 관리하기 위해서는 James에서 메일을 구축할 때, config.xml파일을 수정했던 것에서 추가적으로 문서의 내용을 수정 해 주어야 한다.  기존의 James환경에서는 메일을 보내고 받으면 해당 파일이 파일로 관리되기 때문에, 파일을 관리하기 까다롭기 때문에 Mysql을 사용하여 DB를 통해 관리한다면 더욱 편리하게 메일서버를 관리 할 수 있다.  Mysql을 5.6을 설치하는 과정에서 계정을 추가해주고 이계정을 통해서 config.xml문서를 수정할때 사용하도록 한다.   |  | | --- | | 그림 4 path변경 |   위의 그림 4 와 같이 기존에 Path가 <file://var> .....로 설정되어 있는 것들을 아래에 추가한 코드인db://maildb…..와 같이 바꿔주게 되면 기존에 메일파일이 저장되는 방식이 DB의 경로에 저장이 되게된다. 저 부분 외에도 <file://var>....경로로 설정되어 있는 코드들을 모두 주석처리 또는 삭제를 하고 경로가 db://maildb/,,,,로 향하도록 설정해 주면 된다.  그리고 MySql과 연동하기 위해서 config.xml 파일에 추가적으로 작성해야 하는 소스코드는 아래와 같다.   |  | | --- | | <data-source name="maildb" class="org.apache.james.util.dbcp.JdbcDataSource">  <driver>com.mysql.jdbc.Driver</driver>  <dburl>jdbc:mysql://127.0.0.1/test?autoReconnect=true</dburl>//생성한 DB명 경로 설정  <user>yjg</user>//DB설치시 추가한 계정명  <password>yjg1004</password>//DB설치시 추가한 계정의 Password  <max>20</max>  </data-source> |   위 소스코드는 DB와 연동하기 위해서 필요한 라이브러리를 사용하는 내용을 포함하고 있다. 그리고 3번재 line의 ip주소 다음의 test는 Mysql에서 생성한 Database의 이름이다. User와 password는 초기 Mysql설치 시 설정하였단 계정과 password를 입력해서 DB에 접근이 가능하도록 구현하였다.  **JSP를 이용한 메일 관리 인터페이스 구현**  메일 서버 웹 페이지(JSP)에서 서버에 등록된 사용자로 로그인후 해당 사용자의 메일함을 확인할 수 있고 이를 삭제할 수 있다.  계정 생성시에 계정의 ID를 UTF-8로 인코딩 된 형식으로 저장하며, 화면에 보여줄 때는 이를 다시 디코딩하여보여준다. 인코딩 및 디코딩에 대한 부분은 보고서 내의 다른 부분에서 다루기로 함. 구체적으로 메일 리스트를 표현하는 부분은 코드를 통해 설명하겠다  Properties properties = **new** Properties();  properties.put("mail.pop3.host",mailPop3Host);  SessionemailSession = Session.getDefaultInstance(properties);  POP3StoreemailStore = (POP3Store) emailSession.getStore(mailStoreType);  emailStore.connect(mailUser, mailPassword);  위의 코드를 통하여 POP3의 편지함에 접속하였으므로 POP3 서버 내의 받은 메일들이 저장된 공간 (여기서는 Folder를 사용함.)을 열어야 한다. 그에 해당하는 코드는 아래와 같다.  FolderemailFolder = emailStore.getFolder("INBOX");  emailFolder.open(Folder.READ\_ONLY);  POP3 서버 내의 메일함에 접근하였으므로, 메일의 내용을 모두 뽑아오는 코드를 작성하면 된다. 코드는 아래와 같다.  Message[] messages = emailFolder.getMessages();  **for** (**int**i = 0; i<messages.length; i++) {  Messagemessage = messages[i];  out.println("Subject: " + message.getSubject()+"<br>");  out.println("Content Type: "+ message.getContentType()+"<br>");  out.println("From: " + URLDecoder.decode(message.getFrom()[0].toString(),"UTF-8")+"<br>");  out.println("Text: "+ message.getContent().toString()+"<br><br>");  }  위의 예시에서는 발신인, 제목, Content-Type 및 내용만을 가져오는 것으로 작성이 되어 있으나, message의 get- 메소드를 통해 메일의 다른 내용들도 가져올 수 있다. 여기서 Message 객체는 E-Mail 메시지를 표현한 것으로, E-mail 양식에 필요한 모든 내용이 들어있다.  **Encoding/Decoding을 통한 UTF-8 계정 지원 기능**    base64 encoding 기법으로 local part 부분을 utf-8코드로 인코딩 하는 부분  SMTP 서버에 접속해 메일을 전송할수 있는 sendmail 클라이언트를 작성하였는데 javax.mail.internet 패키지를 사용하여 메일서버를 구축하였다  이 패키지를 이용하여 메일을 전송할 시에 local 파트에 아스키외에 문자, 즉 한글이 올경우 javax.mail.internet.addressexception 에러가 발생하여 메일 전송이 되질 않는다.  이를 해결하기 위해 로컬과 도메인 파트를 따로 떼어낸 뒤에 로컬 파트에 모종의 인코딩 기법을 사용하여 이 부분을 코드화 한 후에 전송하여야 한다. 메세지 양식에서 이러한 인코딩을 적용하는 방법으로 base64 Encoding이라는 방식이 있으며, 이는 RFC4648 (<http://tools.ietf.org/html/rfc4648>) 의 표준을 따른다. 이를 메시지 양식 (여기서는 MIME Message 방식을 사용하며, 이는 RFC2047의 표준을 따른다. <http://tools.ietf.org/html/rfc2047> ) 에 적용하면 아래와 같은 형태로 나타나게 된다.  ex) 권상혁@localhost - > =?UTF-8?B? 6raM7IOB7ZiB?=@localhost  일반적으로 MIME Message에서는 =으로 시작과 끝을 구분하고, ?를 경계로 하여 각각 코드, 인코딩 방식, 인코드된 로컬 파트로 나뉘게 된다. 즉 위에 따르면 UTF-8은 인코드될 코드명을 말하고, B는 base64 방식으로 인코드 시킴을 이야기한다.  이외에 해결 방안으로 퓨니 코드(Punycode) 라는 것이 있는데, 이는 Unicode를 ASCII 기반의 텍스트로 인코딩하는 방법을 나타낸 것으로, 유니코드가 지원하는 모든 언어로 국제화 도메인을 쓸 수 있게 한 IDNA의 일부로서 설명되고 있는데, 이를 local part에 적용시켜 다국어 계정 간 이메일 송수신이 가능하게 할 수 있다. 여기서 지원하는 퓨니코드는 앞에 ‘xn—‘이 붙게 되는데 다음과 같은 형태로 변환된다.  [테스트1@utf8.iptime.org](mailto:테스트1@utf8.iptime.org) -> [xn--1-9e5f988awlb@utf8.iptime.org](mailto:xn--1-9e5f988awlb@utf8.iptime.org)  즉 Non-ASCII 로 된 local part를 ASCII 문자로 코드화 시킴으로서, Non-ASCII 문자를 허용하지 않는 SMTP 서버에서도 다국어 계정 간 메일 송수신이 가능하게 할 수 있다. 다만 이 경우 양 측 서버가 모두 퓨니코드를 지원해야 할 것이다.  **SendMail의 인터페이스 구현**   1. MailForm 구현   ->메일보내기창 삽입    그림 6 메일이 정상적으로 전송되었을 경우의 메시지   1. 소스코드     그림 7 Mail Form 소스코드  **메일 보관함에서의 메일 삭제 기능**  1) mailDelete의 구현    Jsp의 체크 박스를 사용하여 폴더안에 있는 메일(DB)를 선택적으로 삭제할 수 있게  구현  2) 소스코드 |

**JSP를 이용하여 MySQL에 접속해 계정 생성하여 DB에 저장**

1) 브라우저상에 구현된 계정생성 FORM



2) 소스코드

**보안을 위한 사용자 비밀번호의 암호화**

1) SHA 암호화 방식을 선택한다

md = MessageDigest.getInstance("SHA");

2) 인코딩하여 암호화 한다

**byte**[] digest = md.digest(pwdHash.getBytes("iso-8859-1"));

bos = **new** ByteArrayOutputStream();

OutputStream encodedStream = MimeUtility.encode(bos,"base64");

3) 암호화된 비밀번호를 DB에 저장하기 위해 스트링화 한다

encodedStream.write(digest);

pwdHash = bos.toString("iso-8859-1");

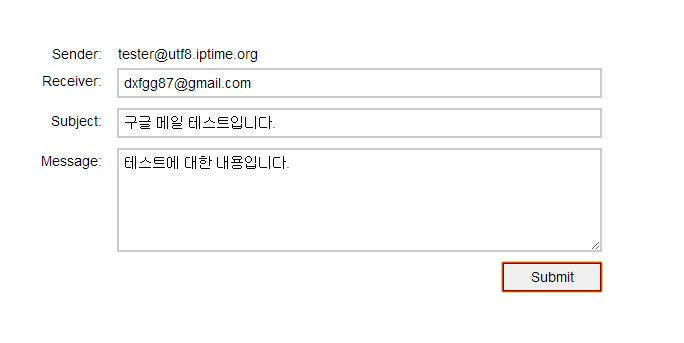
1. **과제 수행 내용 및 최종결과**
2. local 환경에서 메일 송수신

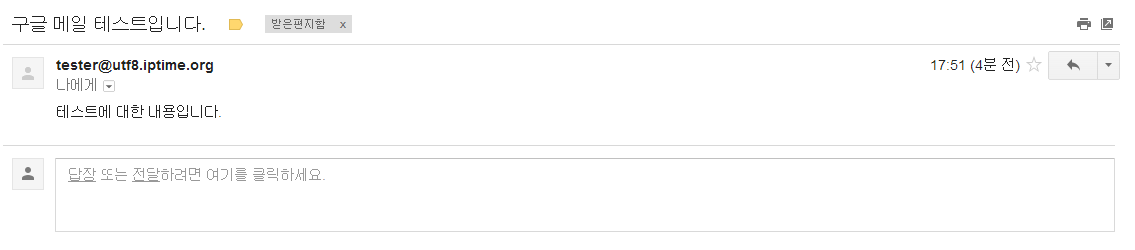
|  |
| --- |
|  |
| 1. 서버를 설치할 PC의 IP를 James 의 config파일에 적용시킨뒤 서버를 실행한다. |
| 1. 웹브라우저를 통해 메일서버에 접속한뒤 테스트를 진행한다   C:\Users\weefrill\Desktop\최종\login.png |
|  |
| 1. 계정을 한글로 생성한다   C:\Users\weefrill\Desktop\최종\create.png   1. 로그인 성공   C:\Users\weefrill\Desktop\최종\logcom.png   1. 권상혁 계정으로 메일을 작성하여 전송한다.      1. 전송 완료   C:\Users\weefrill\Desktop\최종\sendtestcom.png   1. 윤정용 계정으로 다시 로그인하여 메일을 수신을 TEST      1. 메일 내용을 확인하여 정상적인 내용을 출력하는지 확인 |

1. 외부서버와의 메일송수신

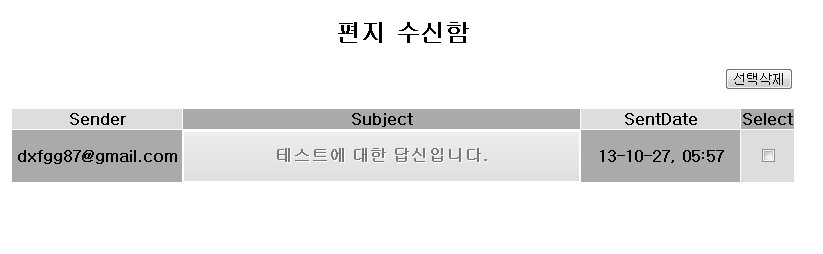
외부메일과의 통신을 위해 현재 소스코드에서는 다운그레이드를 지원하지 않으므로, ASCII형식의 아이디를 추가로 생성 한 뒤에 테스트를 하였다.

1. gmail로 메일을 송신 테스트





1. gmail로부터 메일 수신 테스트





성공적으로 utf8.iptime.org 도메인 주소로 메일이 도착했음을 확인 할 수 있다.

1. 최종결과

* WIndows환경에서 java의 오픈소스 메일서버인 james 2.3.2를 통해서 메일서버 구축
* MySql(DB)와의 연동을 통해서 사용자의 계정정보 관리 및 송수신한 메일의 데이터를 효과적으로 관리할 수 있도록 구현
* JSP를 사용하여 메일서비스를 웹 브라우저 상에서 이용할 수있는 인터페이스를 구현하여 Test
* 현재 구현한 서버를 도메인으로 사용하는 사용자끼리는 메일을 주고 받을 수 있게 구현된 상태이며, gmail같은 외부 메일 서비스와 의 송수신도 원활하게 되고 있음을 확인하였다.

1. **추진 체계**

|  |  |  |
| --- | --- | --- |
| **구성원** | 역 할 | 공통 역할 |
| **송 영찬** | **Java 및 JSP를 사용하여 Web페이지에서 메일**  **송신기능 구현** | **James – 메일 서버환경 구현**  **보고서 수정 및 검토** |
| **염 재걸** | **구현된 메일서버를 MYSQL(데이터베이스)에 연동 및 TEST** |
| **윤 정용** | **Java,JSP를 사용하여 Web페이지에서 수신 메일함 구현 및 테스트** |
| **권 상혁** | **Encoding/Decoding을 통한 UTF-8 계정 지원 기능** |

1. **추진 일정**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **내용** | **6주차(7월)** | **7주차(7/8월)** | **8주차(8월)** | **9주차(8월)** | **10주차(8월)** | **11주차(8/9월)** | **12주차(9월)** |
| **설계** | 메일 시스템 설계 |  |  |  |  |  |  |
| UTF8 인코딩설계 |  |  |  |  |  |  |
| 메일 서비스 인터페이스 설계 |  |  |  |  |  |  |
| **구현** |  | James SMTP 서버통신부분구현 |  |  |  |  |  |
|  | James SMTP Server / Client 연동 |  |  |  |  |  |
|  | 메일서버 Test 및 DB연동 | |  |  |  |  |
|  |  | JSP를사용한 Web Mail 페이지구현 | |  |  |  |
| **테스트** |  |  |  |  | Local서버상의 송수신 테스트 | |  |
|  |  |  |  | 외부 서버와의 송수신 테스트 | |  |
| **문서화** |  |  | 중간보고서 |  |  |  | 최종보고서 |