ApacheCon North America 2011

4 Mailbox Technologies for James

Eric Charles, U-Mangate
eric@apache.org, Nov 9 2011

Presented by

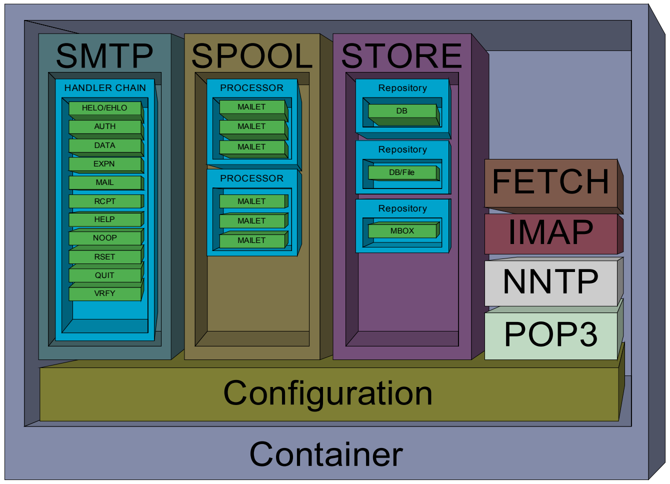
The Apache Software Foundation
http://www.apache.org/

Produced by

STONE CIRCLE PRODUCTIONS, LLC

What is James

“The Apache James Project delivers a rich set of open source modules and libraries, written in Java, related to Internet mail communication which build into an advanced enterprise mail server”



http://james.apache.org

http://blogs.apache.org/james

http://twitter.com/ApacheJames

James Components

James is divided in projects, so as they can be used in other products easily:

- **Server**: Enterprise mail server, it depends on other modules to be built.
- **Protocols**: Framework for mail protocols implementations.
- **IMAP**: Flexible codec for IMAP and command processors.
- **Mailbox**: A library providing a flexible Mailbox storage accessible via JAVA API.
- **Mime4J**: Parser for mime messages like javax.mail but more tolerant.
- **Mailets**: Email processing engine, it defines and implements a Mailet-API.
- **JSieve**: Java Framework implementation of the Sieve mail filtering language.
- **JSPF**: Java Sender Policy Framework implementation.
- **JDKIM**: Java Framework Implementation for DomainsKeys identification Mail.
- **HUPA**: Hupa is an IMAP-based RIA Webmail written in GWT.
- **MPT**: Mail Protocol Tester (testing of ASCII based line protocols).
- **PostAge**: application for generating mail traffic on mail servers.

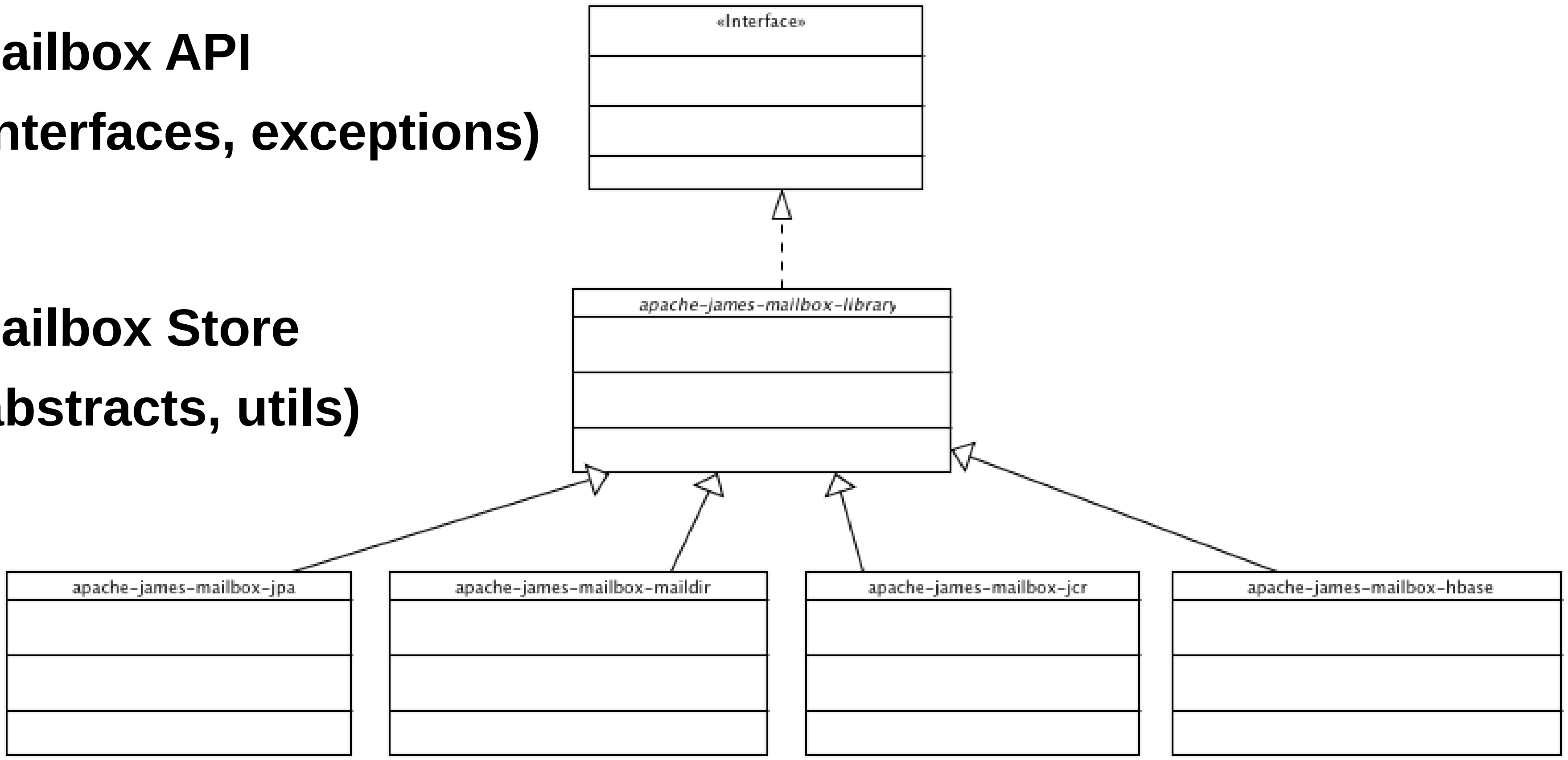
Mailbox Modules

Mailbox API

(interfaces, exceptions)

Mailbox Store

(abstracts, utils)


«Interface»
apache-james-mailbox-library
apache-james-mailbox-jpa
apache-james-mailbox-maildir
apache-james-mailbox-jcr
apache-james-mailbox-hbase

4 different implementations

http://svn.apache.org/repos/asf/james/mailbox/trunk/

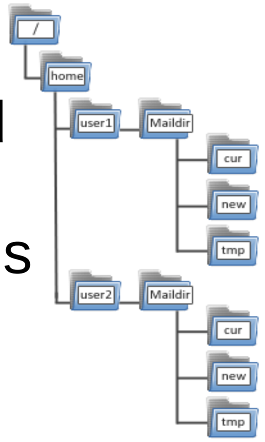
Currently release 0.3

Mailbox Implementations

SQL Database with OpenJPA 2.1

Default implementation

Maildir standard
(think qmail...)
Disk based mails



Mailboxes

JCR (Java Content Repository)

Distributed Mailboxes

(*very first early version*)

Use Mailbox in your application

```
<dependency>
    <groupId>org.apache.james</groupId>
    <artifactId>apache-james-mailbox-jpa</artifactId>
    <version>0.3</version>
</dependency>
```

```
MailboxSession session =
            getMailboxManager().createSystemSession("USER_1");
getMailboxManager().createMailbox(MailboxPath.inbox(session), session);
MailboxPath inboxSubMailbox = new MailboxPath(inbox, "INBOX.Test");
getMailboxManager().createMailbox(inboxSubMailbox, session);
...
getMailboxManager().deleteMailbox(inboxSubMailbox, session);
getMailboxManager().logout(session, false);
```

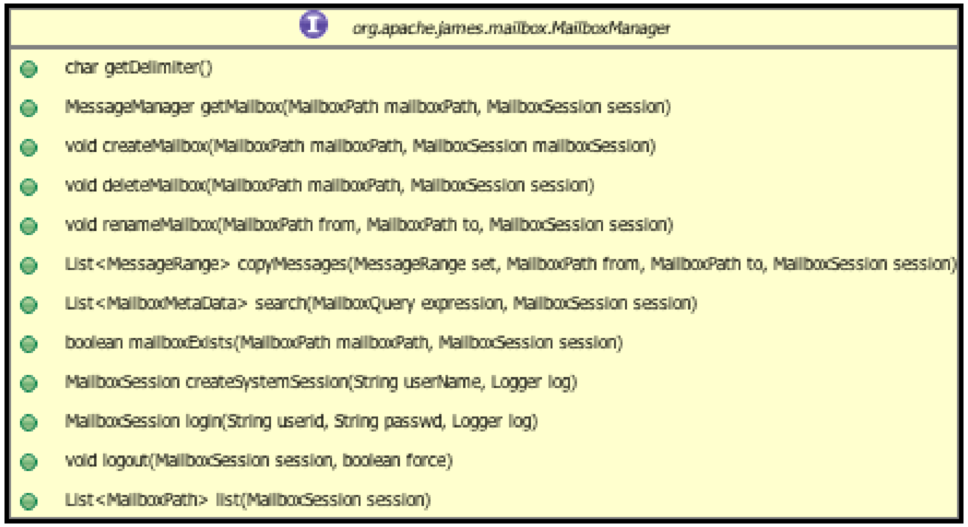
Use it as the "Mailbox pattern", see e.g.
http://www.mindspring.com/~mgrand/pattern_synopses3.htm#Mailbox

Implement a new Mailbox technology

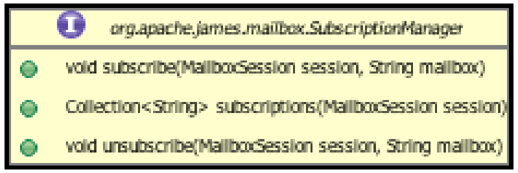
- Rely on API and Store modules
  - Reuse common stuff
  - Direct connection from SMTP/POP3/IMAP4
  - Data Model imposed by the Store

Implement Managers

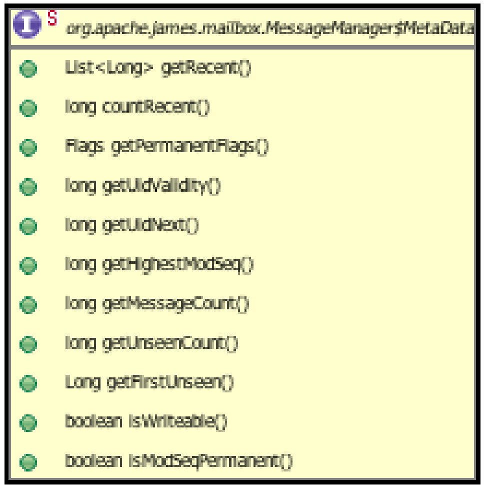
- Mailbox, Message and Subscription Managers
  - mailbox-store will inject your Mappers (Transactional on NonTransactional)



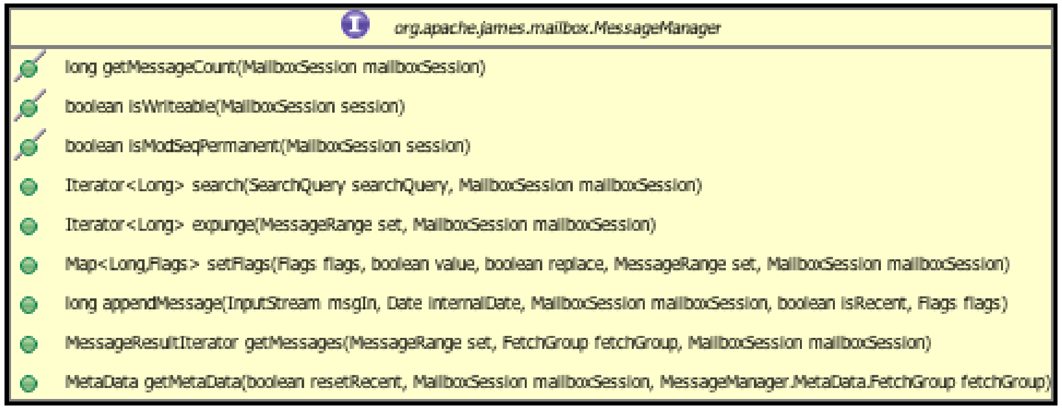
```
org.apache.james.mailbox.MailboxManager
char getDelimiter()
MessageManager getMailbox(MailboxPath mailboxPath, MailboxSession session)
void createMailbox(MailboxPath mailboxPath, MailboxSession mailboxSession)
void deleteMailbox(MailboxPath mailboxPath, MailboxSession session)
void renameMailbox(MailboxPath from, MailboxPath to, MailboxSession session)
List<MessageRange> copyMessages(MessageRange set, MailboxPath from, MailboxPath to, MailboxSession session)
List<MailboxMetaData> search(MailboxQuery expression, MailboxSession session)
boolean mailboxExists(MailboxPath mailboxPath, MailboxSession session)
MailboxSession createSystemSession(String userName, Logger log)
MailboxSession login(String userid, String passwd, Logger log)
void logout(MailboxSession session, boolean force)
List<MailboxPath> list(MailboxSession session)
```



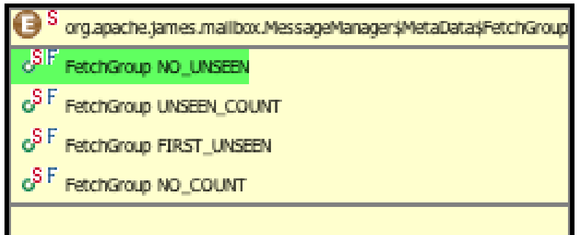
```
org.apache.james.mailbox.SubscriptionManager
void subscribe(MailboxSession session, String mailbox)
Collection<String> subscriptions(MailboxSession session)
void unsubscribe(MailboxSession session, String mailbox)
```



```
org.apache.james.mailbox.MessageManager$MetaData
List<Long> getRecent()
long countRecent()
Flags getPermanentFlags()
long getUidValidity()
long getUidNext()
long getHighestModSeq()
long getMessageCount()
long getUnseenCount()
Long getFirstUnseen()
boolean isWriteable()
boolean isModSeqPermanent()
```



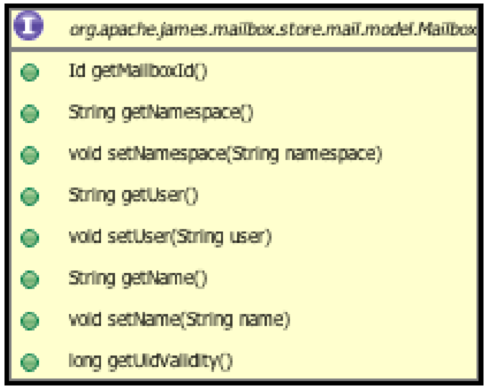
```
org.apache.james.mailbox.MessageManager
long getMessageCount(MailboxSession mailboxSession)
boolean isWriteable(MailboxSession session)
boolean isModSeqPermanent(MailboxSession session)
Iterator<Long> search(SearchQuery searchQuery, MailboxSession mailboxSession)
Iterator<Long> expunge(MessageRange set, MailboxSession mailboxSession)
Map<Long,Flags> setFlags(Flags flags, boolean value, boolean replace, MessageRange set, MailboxSession mailboxSession)
long appendMessage(InputStream msgIn, Date internalDate, MailboxSession mailboxSession, boolean isRecent, Flags flags)
MessageResultIterator getMessages(MessageRange set, FetchGroup fetchGroup, MailboxSession mailboxSession)
MetaData getMetaData(boolean resetRecent, MailboxSession mailboxSession, MessageManager.MetaData.FetchGroup fetchGroup)
```



```
org.apache.james.mailbox.MessageManager$MetaData$FetchGroup
FetchGroup NO_UNSEEN
FetchGroup UNSEEN_COUNT
FetchGroup FIRST_UNSEEN
FetchGroup NO_COUNT
```

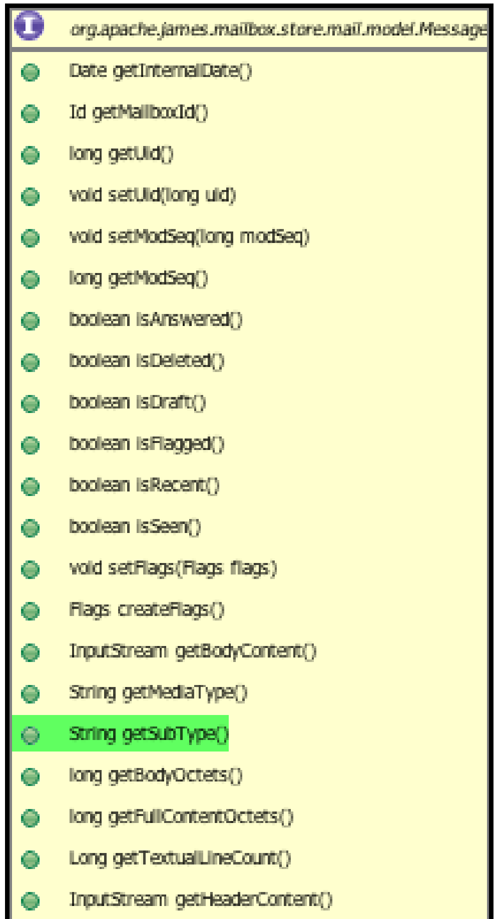
Adapt the Data Model

- Implement the Mailbox, Message and Property interfaces
- User Management out of Mailbox



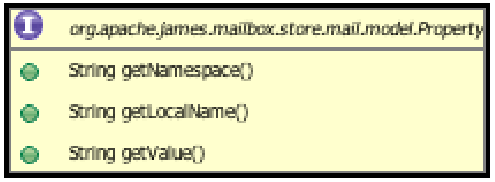
org.apache.james.mailbox.store.mail.model.Mailbox

```
Id getMailboxId()
String getNamespace()
void setNamespace(String namespace)
String getUser()
void setUser(String user)
String getName()
void setName(String name)
long getUidValidity()
```



org.apache.james.mailbox.store.mail.model.Message

```
Date getInternalDate()
Id getMailboxId()
long getUid()
void setUid(long uid)
void setModSeq(long modSeq)
long getModSeq()
boolean isAnswered()
boolean isDeleted()
boolean isDraft()
boolean isFlagged()
boolean isRecent()
boolean isSeen()
void setFlags(Flags flags)
Flags createFlags()
InputStream getBodyContent()
String getMediaType()
String getSubType()
long getBodyOctets()
long getFullContentOctets()
Long getTextualLineCount()
InputStream getHeaderContent()
```



org.apache.james.mailbox.store.mail.model.Property

```
String getNamespace()
String getLocalName()
String getValue()
```

- **Thank You!**
- **Questions?**

Eric Charles, James PMC

eric@apache.org

@echarles

http://about.echarles.net