**Отчёт обучающегося по практике**

Обучающийся: Мясников Максим Александрович Группа: ПИН-44

Направление: Программная инженерия

Образовательная программа:09.03.04 Программные технологии распределенной обработки информации

Вид практики: учебная ознакомительная практика

{Весенний/Осенний} семестр 2021/2022 учебного года

Место практики: ООО «Кодэстетик» (до 27.04.2021 ООО «АйТиГлобал»)

Руководитель практики от МИЭТ / Гагарина Л. Г. /

Руководитель практики   
от организации / Капитанов А. Е. /

Обучающийся / Мясников М. А. /

Введение

В современном мире сложно представить такого человека, который бы не имел бы своего электронного почтового ящика. Более того, люди имеют в своем распоряжении по нескольку электронных почтовых адресов, которые могут относиться к одному или к разным почтовым серверам. Во втором случае доступ к письмам, хранящимся на сервере, обычно осуществляется с помощью сервисов, предоставляемых теми же компаниями, которые являются и владельцами сервера, к которому относится адрес электронной почты. Так, для сервера yandex.ru используется сервис «Яндекс.Почта», для mail.ru, inbox.ru, bk.ru – «Почта Mail.ru», для gmail.com – «Gmail», для outlook.com – «Outlook.com» и т. п.

При наличии нескольких электронных почтовых адресов, их обход и проверка всех почтовых ящиков на наличие новых писем, может превратиться в сложную задачу. Также обладатель нескольких почтовых адресов, может столкнуться с проблемой, поиска письма, принадлежность которого к какому-либо из почтовых ящиков он забыл. В таком случае ему придется проверять всех их в поисках потерянного письма, что может быть довольно-таки утомительным процессом.

Для решения этих проблем, хотелось бы иметь единый доступ к имеющимся электронным почтовым ящикам.

1. Актуальность предметной области

Электронная почта является востребованной технологией, позволяющей вести переписку (личную, деловую), передавать как текстовые сообщения, так файлы различных форматов; служащей для регистрации на различных сервисах.

Наличие у человека нескольких адресов электронной почты является распространённым явлением. Для облегчения доступа к различным электронным почтовым ящикам существуют различные решения: почтовые сервисы (упомянутые выше «Яндекс.Почта», «Почта Mail.ru», «Gmail» и т. д.) и настольные почтовые клиенты (такие как «Microsoft Outlook», «Mozilla Thunderbird», «The Bat!», «eM Client», «Mailbird» и т. п.)

--Чета обосрать—

Поэтому принято решение разработать веб-службу с открытым интерфейсом для доступа к различным почтовым ящикам. В дальнейшем веб-службу можно будет использовать как самостоятельно, так и в силу открытого интерфейса встраивать в другие приложения.

1. Изучение предметной области
   1. Протоколы работы с электронной почты

На данный момент основными протоколами для работы с электронной почтой являются POP3 и IMAP для получения электронных писем с почтового сервера, и SMTP для отправки писем на почтовый сервер.

POP3 (Post Office Protocol Version 3) – интернет-протокол прикладного уровня, используемый для получения электронной почты с удаленного почтового сервера. Принцип работы с протоколом заключается в проверке удаленного сервера на наличие новых писем, последующего их скачивания на устройство пользователя и удаление скачанных писем с сервера. При этом письмо скачивается целиком, со всеми вложениями. К преимуществам такого подхода получения писем можно отнести:

* экономия трафика, так как письма загружаются в локальное хранилище один раз, и хранятся на нем;
* быстрый доступ к вложениям, т.к. они загружаются вместе с письмами;
* сохранение незаполненным дискового пространства почтового сервера, выделенного под почтовый ящик.

Однако, также можно выделить и ряд недостатков данного подхода:

* риск потери данных при поломке локального носителя, на котором хранятся загруженные письма;
* невозможность синхронизации между несколькими компьютерами.

Таким образом, данный подход подошел бы пользователям, которым достаточно иметь доступ к электронной почте с одного устройства, при условии, что они будут периодически выполнять резервирование данных, с целью снижения риска потери загруженных писем.

IMAP (Internet Message Access Protocol) – интернет-протокол прикладного уровня, также как и POP3, служащий для получения доступа к электронным письмам на удаленном почтовом сервере. Разработанный как альтернатива протоколу POP3, имеет ряд доработок, связанных с возможность работы с почтой непосредственно на сервере, без загрузки их на локальное устройство. Так, к преимуществам протокола IMAP перед POP3 можно отнести:

* хранение писем на почтовом сервере, а как следствие меньший риск их потерять;
* возможность одновременного доступа к письмам с нескольких устройств;
* установка флагов письмам (прочитано, отвечено, черновик и т. п.).

Переход к постоянному хранению писем на почтовом сервере, помимо преимуществ имеет и несколько недостатков:

* необходимость постоянного интернет-соединения для доступа к почтовому ящику;
* хоть риск потери электронных писем и ниже, чем при работе с POP3 и хранением писем на локальном устройстве, вероятность поломки серверного оборудования также присутствует.

Исходя из выше написанного следует что протокол IMAP является более универсальным по сравнению с POP3. Если у пользователя нет проблемы с доступом в интернет и получение писем не ограничено одним единственным устройством, то следует использовать именно IMAP.

* 1. Подходы к построению API веб-служб

Основными подходами к построению API веб-служб являются RESTful API и SOAP API. Если первый подход основывается на архитектурном стиле REST (Representational State Transfer – передача репрезентативного состояния), то второй – на протоколе обмена сообщениями SOAP (Simple Object Access Protocol – простой протокол доступа к объекту). Оба подхода имеют существенные различия.

Рассмотрим, что из себя представляет протокол SOAP. Сообщение по протоколу SOAP передаются в формате XML, и имеют структуру:

* Envelope – корневой элемент, являющийся обязательным;
* Header – заголовок; необязательный элемент содержащий атрибуты сообщения;
* Body – тело; обязательный элемент содержащий непосредственно сообщение;
* Fault – ошибки; необязательный элемент в котором описываются ошибки.



Рисунок 1.1 Сообщение SOAP-запрос встроенное в HTTP-запрос



Рисунок 1.2 Сообщение SOAP-ответ встроенное в HTTP-запрос

Строгость спецификации SOAP-сообщений, с одной стороны, является их преимуществом, так как представляет собой строгий стандарт, но тем самым усложняет реализацию.

К преимуществам SOAP также относится неразрывно идущий с этим протоколом WSDL (Web Service Description Language) – язык описания веб-сервисов. WSDL как и SOAP имеет XML-синтаксис и полностью описывает функции и процедуры веб-службы и способы доступа к ним.

В отличие от SOAP API подхода, использующего в своей основе протокол обмена сообщений, RESTful API подход основывается на архитектурном стиле REST. Иными словами, здесь нет строго стандарта, но есть набор правил, которым следует придерживаться:

* модель клиент-сервер;
* отсутствие хранения состояния клиента между запросами на сервере;
* кэширование ответов сервера;
* единообразие интерфейса;
* иерархическая структура сетей (слои)
* код по требованию (расширение функциональности клиента за счет получаемых с сервера сценариев)

Вызов функции или процедуры веб-службы при REST подходе представляет собой HTTP-запрос, определяющий к чему обратиться по конечной точке и HTTP методу. Особое внимание стоит уделить использованию методов HTTP – в RESTful API принято использовать:

* GET – для получения ресурса;
* POST – для создания ресурса;
* DELETE – для удаления ресурса;
* PUT – для обновления ресурса.

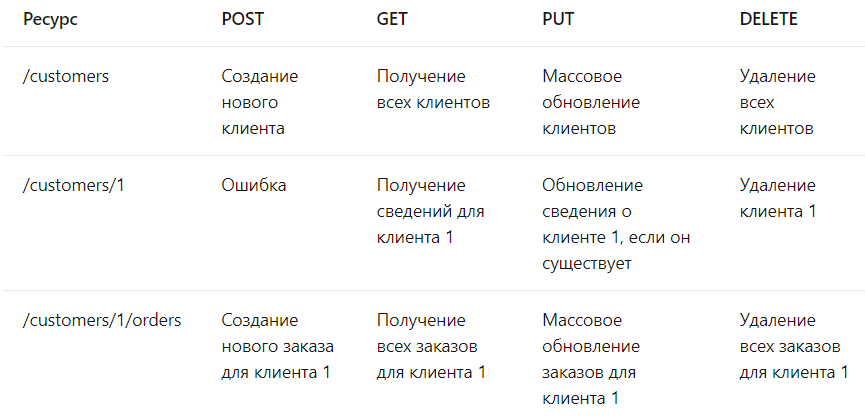


Рисунок 1.3 Пример действий веб-службы для различных HTTP методов и конечных точек.

При этом, в отличие от SOAP, REST не огранивает сообщения определённым форматом представления данных. Хотя наибольшее распространение и имеет формат JSON (который является более гибким и удобочитаемым чем XML), в сообщения могут передаваться и в XML формате, а также в виде обычного текста или бинарных данных и т. п.

Если у SOAP есть WSDL как способ описания, в REST применяется спецификация открытого API (OAS – open API specification), представленная фреймворком Swagger.

Резюмируя,

* 1. Авторизация и аутентификация
  2. Почтовые сервера

Рассмотрим особенности взаимодействия с почтовыми серверами таких компаний как «VK» (до 12 октября 2021 г. «Mail.ru Group»), «Яндекс», «Google» и «Microsoft».

* + 1. Почта Mail.ru («VK»)

Для доступа к почтовому ящику следует обращаться к серверам «imap.mail.ru» (IMAP-сервер) и «pop.mail.ru» (POP3-сервер). В силу использования протокола шифрования SSL/TLS, используемые порты: IMAP – 993, POP3 – 995.

При процессе авторизации, для идентификации и аутентификации используются имя пользователя (полное имя почтового ящика, включая логин, «@» и домен) и пароль (пароль для внешнего приложения). Внешними приложениями считаются все кроме сервисов Mail.ru: Агент, Облако, Почта, Мой Мир.

Для аутентификации через внешнее приложение же необходимо заранее через сервисы Mail.ru создать пароль для внешних приложений. Это может быть как один пароль, так и несколько (рекомендуется использовать разные внешние пароли для разных приложений). Так, если злоумышленник получит доступ к внешнему паролю, к него все равно не будет полномочий ко всем возможностям учетной записи, а только к почтовому ящику. При такой утечке внешнего пароля он просто удаляется в настройках учетной записи.



Рисунок Пароль для внешнего приложения

* + 1. Яндекс.Почта («Яндекс»)

Для доступа к почтовому ящику следует обращаться к серверам «imap.yandex.ru» (IMAP-сервер) и «pop.yandex.ru» (POP3-сервер). В силу использования протокола шифрования SSL/TLS, используемые порты: IMAP – 993, POP3 – 995.

Также как и для доступа к почтовым ящикам mail.ru, здесь используются полное имя почтового ящика и пароль для внешнего приложения. Особое внимание уделяется тому, что при создании пароля внешнего приложения его можно увидеть только один раз. При его потере рекомендуется удалить потерянный и создать новый.



Рисунок Список паролей для внешних приложений

Все созданные пароли внешних приложений сбрасывается если происходит одно из следующих событий:

* смена основного пароля;
* включение/выключение двухфакторной аутентификации;
* восстановление доступа к учетной записи;
* «выход на всех устройствах».

После этого необходимо заново создавать пароли внешних приложений.

* + 1. Gmail («Google»)

Для доступа к почтовому ящику следует обращаться к серверам «imap.gmail.ru» (IMAP-сервер) и «pop.gmail.ru» (POP3-сервер). В силу использования протокола шифрования SSL/TLS, используемые порты: IMAP – 993, POP3 – 995.

Для получения доступа к почтовым ящикам google для идентификации используется полное имя почтового ящика, но в отличие от mail.ru и yandex.ru, здесь нет пароля внешних приложений. Для аутентификации используется пароль Gmail.

Следует отметить, что во взаимодействии с почтовыми серверами «Google» существуют определённые ограничения. Так, превышение лимита по трафику для протокола IMAP (не более 2500 МБ в день на скачивание и не более 500 МБ в день на загрузку) может привести к блокировке учетной записи. Также для одного аккаунта разрешено не более 15 одновременных подключений по протоколу IMAP. По протоколу POP3 можно включить доступ только на одном клиенте.

1. Обзор существующих аналогов

Для проверки актуальности выбранной темы рассмотрим некоторые схожие решения, представленные на рынке.

* 1. Microsoft Graph

«Microsoft Graph» - представляет собой набор API, предоставляющий доступ к данным и средствам искусственного интеллекта в Microsoft 365. Благодаря этому обеспечивается единая модель программируемости, которую можно использовать для доступа к колоссальному объему данных в Microsoft 365, Windows 10 и Enterprise Mobility + Security.

Microsoft Graph предоставляет интерфейсы REST API и клиентские библиотеки для доступа к данным в нижеуказанных облачных службах Майкрософт:

* основные службы Microsoft 365: Bookings, Календарь, Delve, Excel, Обнаружение электронных данных в Центре соответствия требованиям Microsoft 365, Поиск (Майкрософт), OneDrive, OneNote, Outlook/Exchange, Люди (контакты Outlook), Планировщик, SharePoint, Teams, To Do, Рабочая аналитика;
* службы Enterprise Mobility and Security: Advanced Threat Analytics, Расширенная защита от угроз, Azure Active Directory, Identity Manager и Intune;
* службы Windows 10: действия, устройства, уведомления, универсальная печать.
* Dynamics 365 Business Central;

Главное достоинство Microsoft Graph заключается в удобной навигации объектов и связей в разных службах, предоставляемой в единой конечной точке REST Microsoft Graph. Для версии API Microsoft Graph v1.0 конечной точкой является <https://graph.microsoft.com/v1.0>

Microsoft Graph позволяет вашему приложению получать авторизованный доступ к данным почты Outlook в личной или корпоративной учетной записи. Имея соответствующие делегированные разрешения или [разрешения почты приложения](https://docs.microsoft.com/ru-ru/graph/permissions-reference#mail-permissions), приложение может получать доступ к данным почты вошедшего пользователя или любого пользователя в клиенте.

Запросы API почты выполняются от имени [пользователя](https://docs.microsoft.com/ru-ru/graph/api/resources/user?view=graph-rest-1.0), который может определяться свойством пользователя **id** (уникальным GUID), адресом электронной почты или псевдонимом me, обозначающим вошедшего пользователя.

Сообщения электронной почты представлены ресурсом [message](https://docs.microsoft.com/ru-ru/graph/api/resources/message?view=graph-rest-1.0) и упорядочены в почтовой папке [mailFolder](https://docs.microsoft.com/ru-ru/graph/api/resources/mailfolder?view=graph-rest-1.0). Сообщения и почтовые папки определяются свойством **id**, которое можно получить из операций GET.

Служба предоставляет доступ к следующему функционалу:

* Создание, чтение, ответ, пересылка, отправка, обновление или удаление сообщений;
* Предоставление другому пользователю права отправлять сообщения от имени владельца почтового ящика;
* Показ более важных сообщений;
* Запрос сообщений и их получение в папке поиска;
* Получение содержимого MIME сообщения или его вложения;
* Отправка сообщений с содержимым MIME;
* Добавление, получение или удаление вложений сообщения;
* Получение настроек языка и часовых поясов для пользователя;
* Получение или обновление автоматического ответа, языкового стандарта, часового пояса или рабочего времени пользователя;
* Получение подсказок об особом состоянии других пользователей, например об отсутствии на месте;
* Упорядочивание сообщений в иерархии папок почты;
* Классификация сообщений;
* Автоматизация действий, например пересылка определенных входящих сообщений, с помощью правил папки «Входящие»;
* Получение заголовков сообщений Интернета для сообщения;
* Поиск и фильтрация сообщений;
* Получение уведомлений об изменениях сообщений в папке;
* Синхронизация сообщений или иерархии папок почты;
* Получение пользовательских данных приложения в виде заголовков сообщений Интернета для сообщения;
* Добавление пользовательских данных приложения в сообщение с помощью расширений;
* Доступ к пользовательским данным для редко предоставляемых свойств Outlook MAPI.
  1. Gmail API

Gmail API представляет собой набор API предоставляемый сервисом gmail.googleapis.com. Данный сервис позволяет просматривать и управлять данными почтового ящика Gmail, такими как:

* Сообщение – сообщение электронной почты, состоящее из отправителя, получателя, темы и тела письма. Созданное сообщение является неизменяемым.
* Поток сообщений – набор связанных сообщений, формирующих переписку-беседу (когда получатель отвечает на сообщение своим собственным сообщением).
* Метка – механизм организации сообщений и потоков, позволяющий разделить их на категории. Метки делятся на системные («INBOX» – входящие, «TRASH» – корзина, «SPAM» – спам) и пользовательские.
* Черновик – неотправленное сообщение. В отличии от неизменяемых сообщений, черновик до его отправки и преобразования в сообщение может быть изменен.

Возможности, предоставляемые сервисом:

* извлечение данных для чтения, индексирование, резервное копирование;
* автоматическая и программная отправка сообщений;
* миграция учетной записи электронной почты;
* фильтрация и сортировка писем;
* стандартизация подписей электронной почты в организации.

Подробное описание возможностей, предоставляемых сервисом, можно получить по адресу https://developers.google.com/gmail/api/reference/rest.

API для Google использовать [протокол](https://tools.ietf.org/html/rfc6749) Oauth 2.0 для аутентификации и авторизации. Google поддерживает распространенные сценарии Oauth 2.0, например, для веб-сервера, клиентских приложений, установленных приложений и приложений с ограниченным вводом.

Существенным и самым явным недостатком Gmail API является возможность доступа исключительно к учетным записям почтового ящика Gmail.

* 1. Gmail API

Microsoft Graph

Thunderbird WebExtension API

Gmail Api

|  |  |  |  |
| --- | --- | --- | --- |
| Параметры | Microsoft Graph | Gmail API | ВС ДЭП |
| Взаимодействие с различными почтовыми серверами |  | Нет | Да |
| Наличие документации |  | Да | Да |
| Способ авторизации |  | Oauth 2.0 | Oauth 2.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Цель и задачи разработки

Целью данной работы является повышения уровня простоты интеграции функционала доступа к электронной почте в другие приложения.

Задачами разрабатываемой веб-службы являются:

* доступ к электронным письмам посредством протокола POP3:
  + загрузка писем на устройство пользователя;
  + удаление загруженных писем с почтового сервера;
* доступ к электронным письмам посредством протокола IMAP:
  + загрузка писем на устройство пользователя (без вложений, с вложениями);
  + получение информации о письмах без их загрузки;
  + установка флагов сообщений;
  + удаление сообщений;
  + сортировка и фильтрации сообщений при их поиске;
* взаимодействие с часто используемыми почтовыми серверами:
  + mail.ru (imap.mail.ru, pop.mail.ru);
  + yandex.ru (imap.yandex.ru, pop.yandex.ru);
  + gmail.com (pop.gmail.com, imap.gmail.com);
  + outlook.com (outlook.office365.com);
* взаимодействие с иными почтовыми серверами по указанию пользователя;
* обеспечение дополнительной защиты учетных записей пользователей двухфакторной системой аутентификации.

1. Выбор языка и среды программирования

Для разработки ВС ДЭП был проведен анализ языков программирования, их сравнение. После выбора языка программирования также были изучены среды разработки, предназначенные для выбранного языка. Анализ и сравнение языков программирования, а в последствии сред разработки позволяет выбрать оптимальный вариант доя решения поставленных задач.

* 1. Выбор языка программирования

Критериями для выбора языка программирования послужат:

* поддержка объектно-ориентированной парадигмы программирования (ООП);
* поддержка функциональной парадигмы программирования (ФП);
* статическая типизация данных;
* поддержка динамической типизации данных;
* поддержка обработки исключений;
* реализация многопоточности и асинхронности;
* возможность отложенных вычислений;

Наиболее подходящими под эти критерии являются такие языки как Java, C#, JavaScript, Python, PHP, Ruby.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Критерий | Java | C# | JavaScript | Python | PHP | Ruby |
| Поддержка ООП | Да | Да | Да | Да | Да | Да |
| Поддержка ФП | Нет | Да | Да | Да | Да | Да |
| Статическая типизация | Да | Да | Нет | Нет | Нет | Нет |
| Поддержка динамической типизации | Нет | Да | Нет | Нет | Да | Да |
| Обработка исключений | Да | Да | Да | Да | Да | Да |
| Многопоточность | Да | Да |  | Да | Да | Да |
| Асинхронность | Да | Да | Да | Да | Да |  |
| Отложенные вычисления | Нет | Да | Да | Да |  | Да |

Как видно из проведенного анализа, наиболее подходящим по критериям языком программирования для разработки веб-службы является C#.

* 1. Выбор среды программирования

Наиболее распространёнными средствами разработки на языке С# являются Visual Studio, Rider, MonoDevelop и SharpDevelop.

Также как и для языков программирования выделим основные критерии, на которые будем опираться и рассмотрим каждую из сред разработки.

Так, для эффективной разработки наиболее важными являются признаки:

* бесплатность распространения;
* наличие автоматического дополнения кода;
* наличие статистического анализатора кода;
* наличие отладчика;
* возможность проведения тестирования;
* наличие профилировщиков;
* возможность кроссплатформенной сборки;
* возможность создания проектов по шаблонам.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | Visual Studio | Rider | MonoDevelop | SharpDevelop |
| Бесплатность | Да | Бесплатно для студентов | Да | Да |
| Автоматическое дополнение кода | Да | Да | Да | Да |
| Статистический анализатор кода | Да | Да | Да | Да |
| Отладчик | Да | Да | Да | Да |
| Тестирование | Да | Да | Да | Да |
| Профилирование | Да | Необходимы расширения | Нет | Да |
| Шаблоны проектов | Да | Да | Нет | Нет |
| Кроссплатформенная сборка | Да | Да | Да | Нет |

Наиболее подходящей средой разработки является Visual Studio.

Список литературы

* <https://help.mail.ru/mail/mailer/popsmtp>
* <https://help.mail.ru/mail/security/protection/external>
* <https://yandex.ru/support/mail/mail-clients/others.html>
* <https://yandex.ru/support/id/authorization/app-passwords.html>
* <https://support.google.com/mail/answer/7126229?hl=ru>
* <https://support.google.com/mail/answer/7104828>

Список литературы

<https://developer.microsoft.com/ru-ru/graph>

<https://docs.microsoft.com/ru-ru/graph/>

<https://developers.google.com/gmail/api>

<https://www.w3.org/TR/2000/NOTE-SOAP-20000508/>