Технологии и методы программирования (Лекция 25)

- Обработка форм
- Cookies
- Сессии
- Ajax
- Emails

Обработка форм

- Сервлеты позволяют обрабатывать формы с помощью следующих методов:
  - getParameter() – получение значения по имени элемента формы.
  - getParameterValues() – получение значений у параметров, которые возвращают больше одного значения. Например, checkbox.
  - getParameterNames() – Получение списка всех параметров в текущем запросе.

Cookies

- Ку́ки (cookie) — небольшой фрагмент данных, отправленный веб-сервером и хранимый на компьютере пользователя.
- Веб-клиент (веб-браузер) каждый раз при попытке открыть страницу соответствующего сайта пересылает этот фрагмент данных веб-серверу в составе HTTP-запроса.
- Спецификации указывают минимальные объёмы для хранения куки: браузер должен хранить по меньшей мере 300 куки по 4096 байт каждая, и по меньшей мере 20 куки для одного сервера или домена.

Работа куки

- Установка куки
  Пользователь запрашивает страницу и браузер отправляет веб-серверу HTTP-запрос:

```
GET /index.html HTTP/1.1
Host: www.example.org
```

- Сервер отвечает, отправляя запрашиваемую страницу вместе с текстом, содержащим HTTP-ответ:

```
HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=value

<html>...
```

- Браузер запоминает строку name=value (имя = значение) и отправляет её с каждым последующим запросом:

```
GET /spec.html HTTP/1.1
Host: www.example.org
Cookie: name=value
Accept: */*
```

Cookies

- Применяются для сохранения данных на стороне пользователя:
  - аутентификация пользователя;
  - хранения персональных предпочтений и настроек пользователя;
  - отслеживания состояния сеанса пользователя;
  - ведения статистики о пользователях.

Типы куки

- Сессионные куки — куки, которые существуют только во временной памяти, пока пользователь находится на странице веб-сайта. Браузеры обычно удаляют сессионные куки после того, как пользователь закрывает окно браузера.
- Постоянные куки — куки, которые имеют срок действия
- Защищенные куки — куки, которые могут быть переданы только через шифрованное соединение (то есть HTTPS).
- Http Only-куки — куки к которым запрещён доступ с помощью API, например JavaScript.
- Куки сайта — куки, которые можно отправлять только в запросах, исходящих из того же источника, что и целевой домен.
- Сторонние куки — куки, которые появляются, когда веб-страницы содержат контент с внешних веб-сайтов, например рекламные баннеры.
- Супер-куки — это куки-файл с источником домена верхнего уровня (например, .ru) или общедоступным суффиксом (например, .co.uk).
- "Зомби" куки — это куки, который автоматически воссоздается после удаления. Это достигается путем хранения содержимого куки в нескольких местах, таких как общий объект Flash Local, веб-хранилище HTML5 и другие местоположения на стороне клиента и даже на стороне сервера.

Правовое использование куки

- Директива Европейского союза 2002/58/ЕС о конфиденциальности и электронных средствах связи 2002 г. содержит нормы, касающиеся использования куки:
  - пользователю предоставляется информация о том, как куки используются;
  - пользователь имеет возможность отказаться от их использования.
- В 2009 году Директива 2009/136/ЕС ужесточила требования к порядку сбора информации о посетителях сайтов. Согласно новым правилам владельцы сайтов должны получать предварительное согласие посетителей на сбор информации (в том числе куки) и сообщать о действующих на сайте инструментах сбора информации.
- В мае 2018 года в Евросоюзе вступил в силу Общий регламент по защите данных, заменивший действующую Директиву 2002/58/ЕС, относящийся ко всем сайтам, посещаемым из Евросоюза, и приравнивающий большую часть куки к другим персональным данным. Регламент говорит, что достаточно уведомления пользователя об установке куки.

Сессия

- Сессия – соединение между клиентом и сервером, устанавливаемое на определенное время, за которое клиент может отправить на сервер сколько угодно запросов.
- Сессия устанавливается непосредственно между клиентом и Web-сервером.
- Каждый клиент устанавливает с сервером свою собственную сессию.
- Сессии используются для обеспечения хранения данных во время нескольких запросов Web-страницы.

Internationalization (i18n)

- Интернационализация — позволяет адаптировать приложения к использованию разных языков.
- Locale — объект идентифицирующий язык пользователя и регион.

```
Locale locale;

locale = Locale.getDefault();
locale = new Locale("en", "EN");
locale = Locale.ENGLISH;
```

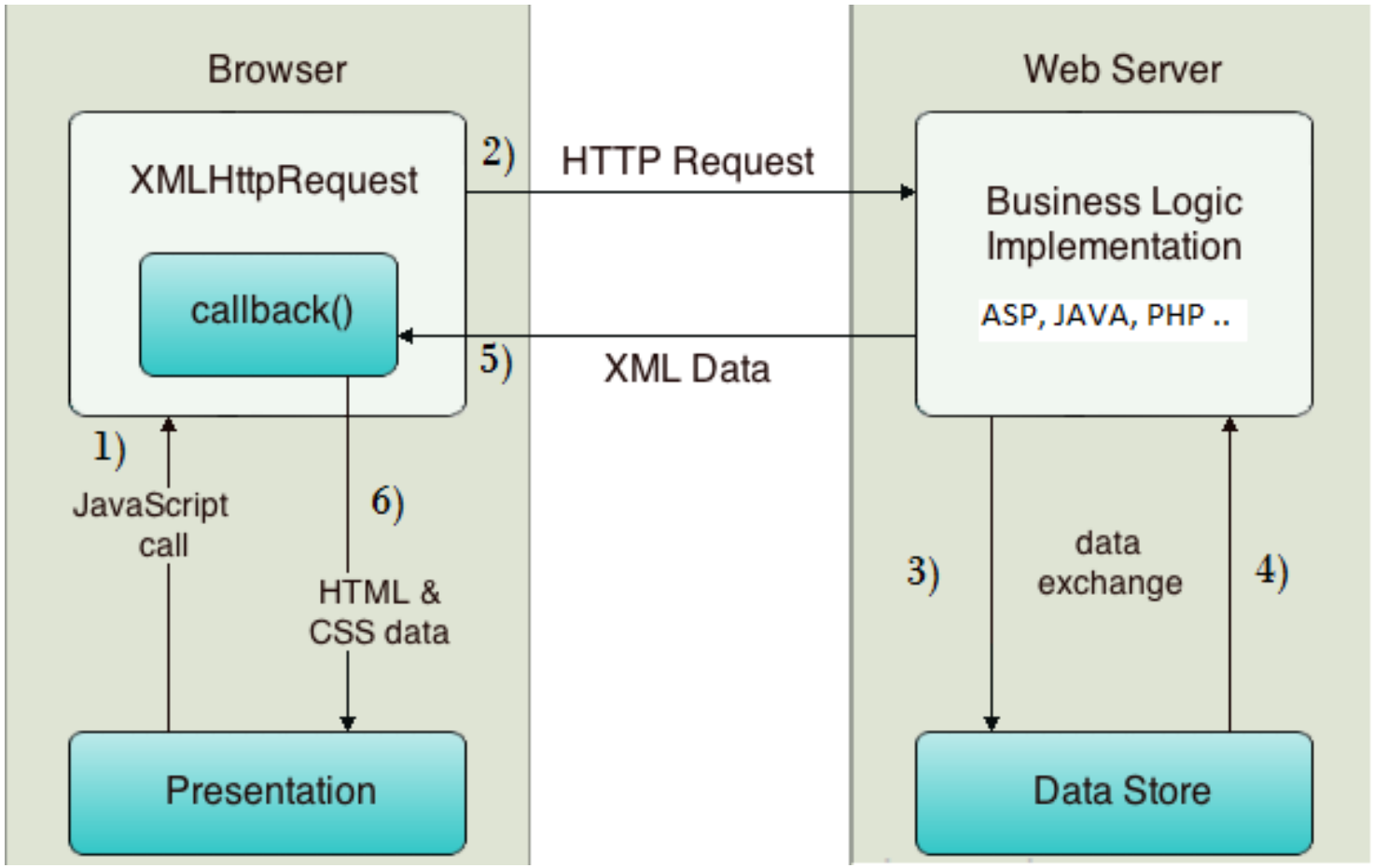
- Файлы переводов *.proprties
- MessagesBundle.properties:

```
greetings = Hello.

farewell = Goodbye.

inquiry = How are you?
```

```
ResourceBundle messages;
messages = ResourceBundle.getBundle("messages", locale);

System.out.println(messages.getString("title"));
```

Asynchronous Javascript and XML

- AJAX — подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером.



Объект XMLHttpRequest

- Объект XMLHttpRequest (или, как его кратко называют, «XHR») дает возможность из JavaScript делать HTTP-запросы к серверу без перезагрузки страницы.
- Несмотря на слово «XML» в названии, XMLHttpRequest может работать с любыми данными, а не только с XML.

Объект XMLHttpRequest

- *Пример асинхронного запроса:*

```
var xhr = new XMLHttpRequest();
xhr.open('GET', 'phones.do', true);

xhr.onreadystatechange = function() {
    if (xhr.readyState != 4)
        return;
    if (xhr.status != 200) {
        alert(xhr.status + ': ' + xhr.statusText);
    } else {
        alert(xhr.responseText);
    }
}
xhr.send();
```

- *Состояния, по спецификации:*

const unsigned short UNSENT = 0; // начальное состояние
const unsigned short OPENED = 1; // вызван open
const unsigned short HEADERS_RECEIVED = 2; // получены заголовки
const unsigned short LOADING = 3; // загружается тело (получен очередной пакет данных)
const unsigned short DONE = 4; // запрос завершён

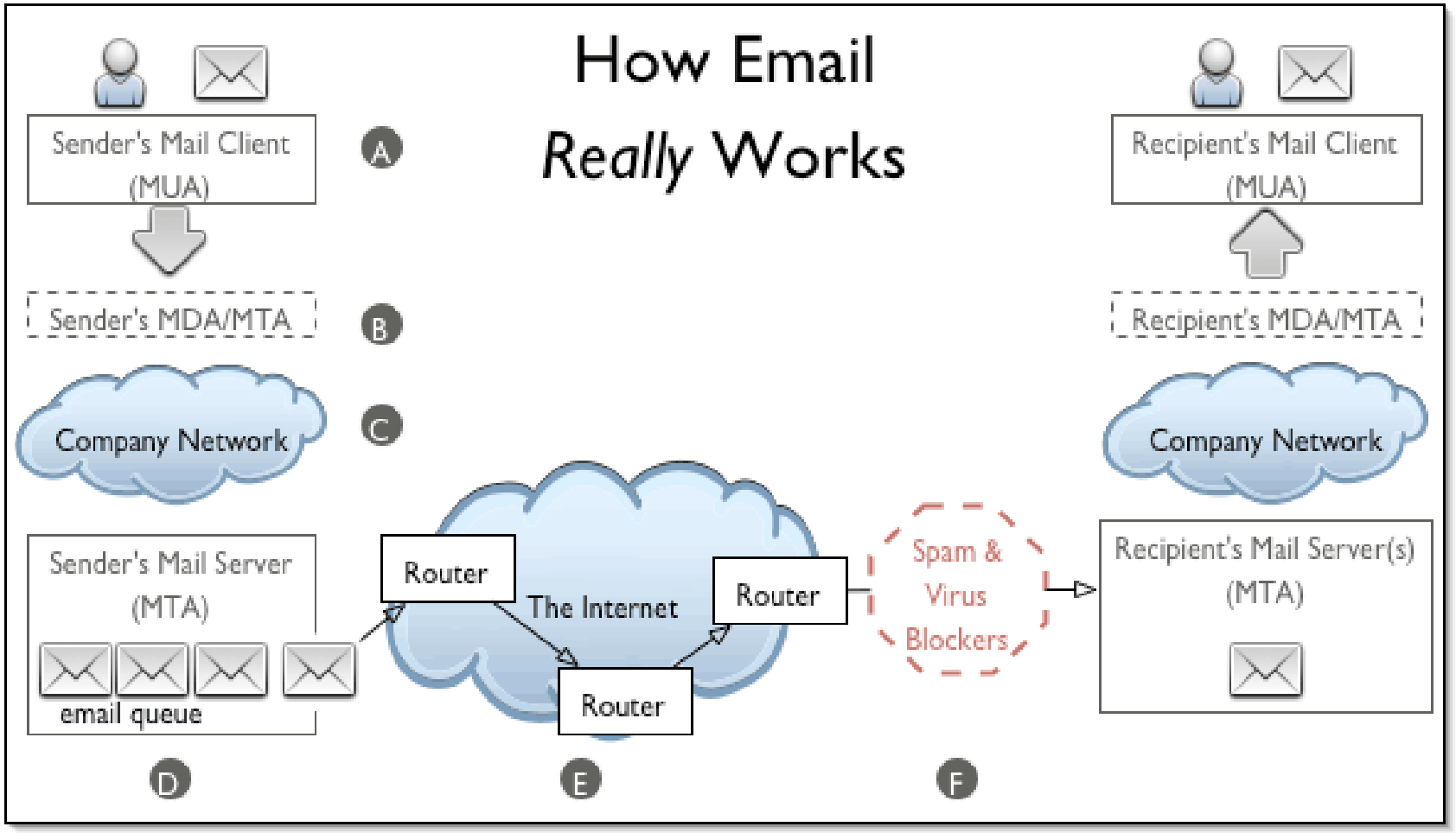
Электронная почта

- Электронная почта — технология и служба по пересылке и получению электронных сообщений между пользователями компьютерной сети (Интернета).
- Адрес электронной почты — запись, установленная по RFC 5322, идентифицирующая почтовый ящик, в который следует доставить сообщение электронной почты.
- Адрес состоит из двух частей, разделённых символом «@». Левая часть указывает имя почтового ящика. Правая часть адреса указывает доменное имя того сервера, на котором расположен почтовый ящик.
- Пример: john.doe@machine.example

Терминология электронной почты

- MTA (Mail Transfer Agent) — отвечает за пересылку почты между почтовыми серверами; как правило, первый MTA в цепочке получает сообщение от MUA, последний передаёт сообщение к MDA.
- MDA (Mail Delivery Agent) — отвечает за доставку почты конечному пользователю.
- MUA (Mail user agent) — программа, обеспечивающая пользовательский интерфейс, отображающая полученные письма и предоставляющая возможность отвечать, создавать, перенаправлять письма.
- MRA (Mail retrieve agent) — почтовый сервер, забирающий почту с другого сервера по протоколам, предназначенным для MDA.

Принцип работы электронной почты



JavaMail

- JavaMail — Java API предназначенное для получения и отправки электронной почты с использованием протоколов SMTP, POP3 и IMAP.
- Существует также альтернативная реализация JavaMail с открытым исходным кодом — GNU JavaMail — которая реализует только спецификацию JavaMail версии 1.3
- В состав JavaMail не входит почтовый сервер.

Пример отправки HTML сообщения

```
String sendFrom = "no-reply@webtechnology.org";
String sendTo = "ivan@gmail.com";

String host = "localhost";
String port = "5225";

Properties properties = System.getProperties();
properties.setProperty("mail.smtp.host", host);
properties.setProperty("mail.smtp.port", port);

Session session = Session.getDefaultInstance(properties);

try {
    MimeMessage message = new MimeMessage(session);
    message.setFrom(new InternetAddress(sendFrom));
    message.addRecipient(Message.RecipientType.TO, new InternetAddress(sendTo));
    message.setSubject("This is the Subject Line!");
    message.setContent("<h1>Message Header</h1>" +
                    "<p>This is a paragraph </p>" +
                    "<p>This is another paragraph</p>", "text/html");

    Transport.send(message);
    System.out.println("Sent message successfully....");
} catch (MessagingException mex) {
    mex.printStackTrace();
}
```

Спасибо за внимание!