Document for PubSubDemo

How to Use:

1. Deploy and run the project on the server
2. Go to http://(server_domain)/index.html
3. Create a new account or login using Username: demo
4. After login you can see the list of subscribed topic for this user, as well as other topics which can be published or subscribed
5. If there are any new message in the subscribed topic, it will highlight the topic name in the list
6. Click the topic, then you can see unread message, request all message or unsubscribe for a subscribed topic; you can also subscribe a new topic after clicking
7. You can publish new message in a topic after clicking it

Design details:

1. Designing the system as a Web application, following the basic Pub/Sub structure
2. Including simple UI for interaction between user and server
3. Using Java (Servlet) as back-end programming while implement front-end using pure HTML and JavaScript
4. The main goal of system is sending and receiving data, which is implemented by communication between user and server under HTTP (GET/POST/PUT) protocol
5. Saving all data in files
6. The whole project was programmed in Eclipse Java EE 4.5.1 and deployed in Tomcat 8.0 for testing, under Mac OS 10.10.5

Programs structure & Function description:

1. **Front-end:**
   a. index.html: user login/registration page
   b. home.html: show the list of user subscribed topics including read (normal) and unread (highlight) topics, and the list of other topics in system
   c. subtopic.html/pubtopic.html: topic details including message and options

d. user.js: implement user login and registration functions, communicates with UserServlet
e. home.js: deal with options in home.html; sending request for topic details to TopicServlet when clicking the topic item
f. topic.js: deal with all pub/sub options

2. **Back-end:**
   a. UserServlet: deal with user creation and validation with GET request; writing to user data when subscribing a new topic with POST request; updating user data after unsubscribing a topic with PUT request
   b. TopicServlet: push unread message or return all topic message with GET request; updating topic data when user publish new message with POST request

Data:

1. All information in system are stored as files
2. Server maintains two file: userlist.txt which records all registered user and topiclist.txt which contains all topic name
3. Each user has a file: (username).txt which contains all topics the user has subscribed
4. Each topic has a file: (topic).txt which contains all messages in this topic

Simplify & limitation:

1. User management just contains username, not password or other information
2. User can publish new message to an existing topic, however, user cannot create new topic
3. Just use files to store data, not using databases which should be more powerful in real case
4. Not using encoding to send, receive and store data, just use the default format
5. UI is really, really simple, just for showing necessary interaction information