

アプリ「どこつぶ」と接続

war ファイルには java ファイルが含まれていないので追加しないと編集はできない
編集後の war ファイルか class ファイルを転送して入れ替えるほうが一般的なのだろうが、
ディレクトリ構成理解も兼ねて、アプリのデプロイ(配置)を手動でやりなおしてから編集してみた

1・アプリ どこつぶ (DocoTsubu) の構成

```
□/opt/tomcat
└─□webapps
   └─■DocoTsubu
      ├──■img
      │   └─┬画像ファイル
      ├──■js
      │   └─┬JavaScript のファイル
      ├──◆docoTsubu_css.css
      ├──◆index.jsp
      └─■WEB-INF
         ├──■lib
         │   ├──◆jstl-api-1.2.jsr
         │   └─┬◆jstl-impl-1.2.jsr
         ├──■jsp
         │   ├──◆loginResult.jsp
         │   ├──◆logout.jsp
         │   ├──◆main.jsp
         │   └─┬◆removeResult.jsp
         ├──■src
         │   ├──■constants
         │   │   └─┬◆MessageConstants.java
         │   ├──■dao
         │   │   └─┬◆MutterDAO.java
         │   ├──■filter
         │   │   └─┬◆DocotsubuFilter.java
         │   ├──■listener
         │   │   └─┬◆Listener.java
         │   ├──■model
         │   │   ├──◆GetMutterListLogic.java
         │   │   ├──◆LoginLogic.java
         │   │   ├──◆MessageBox.java
         │   │   ├──◆Mutter.java
         │   │   ├──◆PostMutterLogic.java
         │   │   └─┬◆User.java
         │   └─┬■servlet
         │       ├──◆Login.java
         │       ├──◆Logout.java
         │       └─┬◆Main.java
         └─■classes
            └─※
```

/opt/tomcat 配下に、
左図の構成で DocoTsubu を配置する

□■ ディレクトリ

◇◆ ファイル

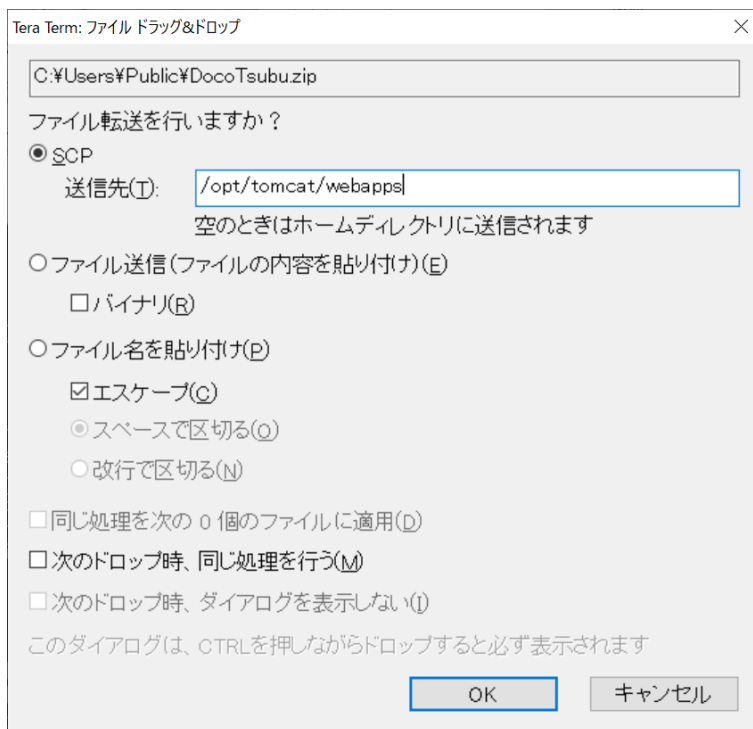
◆ ほぼ入門書通り (加筆あり)

■◆ 独自に追加

JDBC は tomcat/lib に配置されているので
コンテキストには配置しなくても良い

※classes ディレクトリには
java ファイルをコンパイルして
同じ構成で配置していく

2 ・ webapps ディレクトリへ転送



前記のフォルダ構成で
各ファイルを用意して
Zip ファイルにしておく

(classes 内はここでは空で良い)

TeraTerm でその zip ファイルを
webapps ディレクトリへ転送

3 ・ Docotsubu.zip を展開させる

cd /opt/tomcat/webapps; ls #確認

```
[root@localhost ~]# cd /opt/tomcat/webapps; ls #確認
DocoTsubu.zip  bean      dao      examples  manager  testDaoBean
ROOT          classes  docs     host-manager  testApp
```

unzip Docotsubu.zip; ls #.zip を展開して確認

```
[root@localhost webapps]# unzip Docotsubu.zip; ls #.zip を展開して確認
Archive:  Docotsubu.zip
  creating: Docotsubu/
    creating: Docotsubu/WEB-INF/src/servlet/
  inflating: Docotsubu/WEB-INF/src/servlet/Login.java
  inflating: Docotsubu/WEB-INF/src/servlet/Logout.java
  inflating: Docotsubu/WEB-INF/src/servlet/Main.java
Docotsubu      ROOT  classes  docs      host-manager  testApp
DocoTsubu.zip  bean  dao      examples  manager       testDaoBean
```

rm -rf Docotsubu.zip; ls #.zip を削除/確認

```
[root@localhost webapps]# rm -rf Docotsubu.zip; ls #.zip を削除/確認
Docotsubu  bean      dao      examples  manager  testDaoBean
ROOT      classes  docs     host-manager  testApp
```

4 ・ DAO のデータベース接続情報を編集

```
cd DocoTsubu/WEB-INF/src; vi dao/MutterDAO.java #移動/ファイル編集
```

```
[root@localhost webapps]# ¥  
> cd DocoTsubu/WEB-INF/src; vi dao/MutterDAO.java #移動/ファイル編集  
[root@localhost src]#
```

3つの定数を下記のように指定する

```
private final String JDBC_URL = "jdbc:mysql://localhost/docotsubu";
```

```
private final String DB_USER = "root";
```

```
private final String DB_PASS = "AT8_MySQL";
```

```
public class MutterDAO {  
  
    //フィールド(データベース接続に使用する情報)  
    /** JDBCドライバ名 */  
    private final String JDBC_URL = "jdbc:mysql://localhost/docotsubu";  
    private final String DB_USER = "root";  
    private final String DB_PASS = "AT8_MySQL";  
-- INSERT --
```

5 ・ 全ての java ファイルをコンパイルして classes ディレクトリへ配置

: '全てコンパイルしてclassesへ配置' ¥

```
javac -classpath /opt/tomcat/lib/servlet-api.jar */*.java -d ../classes
```

```
[root@localhost src]# : '全てコンパイルしてclassesへ配置' ¥  
> javac -classpath /opt/tomcat/lib/servlet-api.jar */*.java -d ../classes  
[root@localhost src]#
```

ls ../classes/* #classes ディレクトリ内を全て確認

```
[root@localhost src]# ls ../classes/* #classesディレクトリ内を全て確認  
../classes/constants:  
MessageConstants.class  
../classes/dao:  
MutterDAO.class  
../classes/filter:  
DocotsubuFilter.class  
../classes/listener:  
Litener.class  
../classes/model:  
GetMutterListLogic.class  MessageBox.class  PostMutterLogic.class  
LoginLogic.class         Mutter.class      User.class  
../classes/servlet:  
Login.class Logout.class Main.class  
[root@localhost src]#
```

6・再起動させて読み込ませる

```
Systemctl restart tomcat
```

```
[root@localhost src]# systemctl restart tomcat  
[root@localhost src]#
```

7・データベースとテーブルを用意

MySQL にログインし、データベース「docotsubu」とテーブル「MUTTER」を作成

```
mysql> -- データベース作成  
mysql> CREATE DATABASE docotsubu;  
Query OK, 1 row affected (0.01 sec)
```

```
-- データベース作成  
CREATE DATABASE docotsubu;
```

```
mysql> -- データベース選択  
mysql> USE docotsubu;  
Database changed
```

```
-- データベース選択  
USE docotsubu;
```

```
mysql> -- テーブルの作成、フィールドの指定  
mysql> CREATE TABLE MUTTER(  
    -> ID INT PRIMARY KEY AUTO_INCREMENT,  
    -> NAME VARCHAR(100) NOT NULL,  
    -> TEXT VARCHAR(255) NOT NULL,  
    -> DATE VARCHAR(50));  
Query OK, 0 rows affected (0.03 sec)
```

```
-- テーブルの作成、フィールドの指定  
CREATE TABLE MUTTER(  
    ID INT PRIMARY KEY AUTO_INCREMENT,  
    NAME VARCHAR(100) NOT NULL,  
    TEXT VARCHAR(255) NOT NULL,  
    DATE VARCHAR(50));
```

```
mysql> -- テーブルの確認  
mysql> SHOW TABLES;  
+-----+  
| Tables_in_docotsubu |  
+-----+  
| MUTTER               |  
+-----+  
1 row in set (0.00 sec)
```

```
-- テーブルの確認  
SHOW TABLES;
```

```
mysql> -- テーブル内の確認  
mysql> SELECT * FROM MUTTER;  
Empty set (0.00 sec)
```

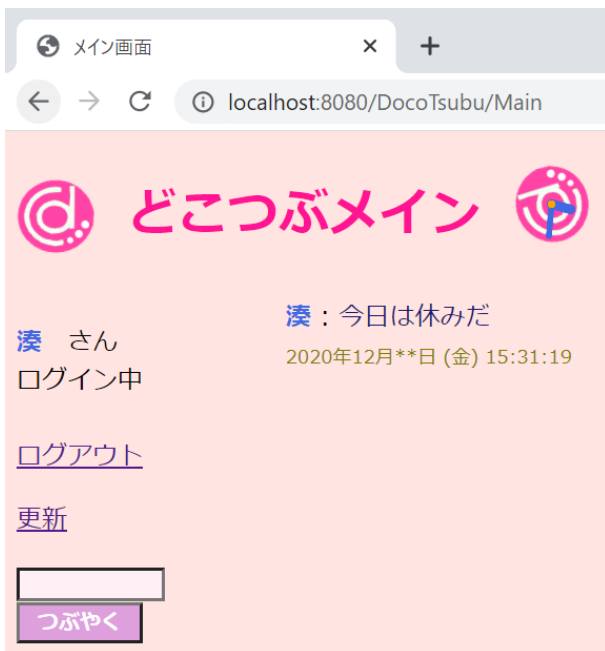
```
-- テーブル内の確認  
SELECT * FROM MUTTER;
```

まだ何もデータが無い

8・どこつぶにログインして接続を確認



1. 「<http://localhost:8080/DocoTsubu/>」
ユーザー「湊」でログイン
2. 「今日は休みだ」とつぶやく



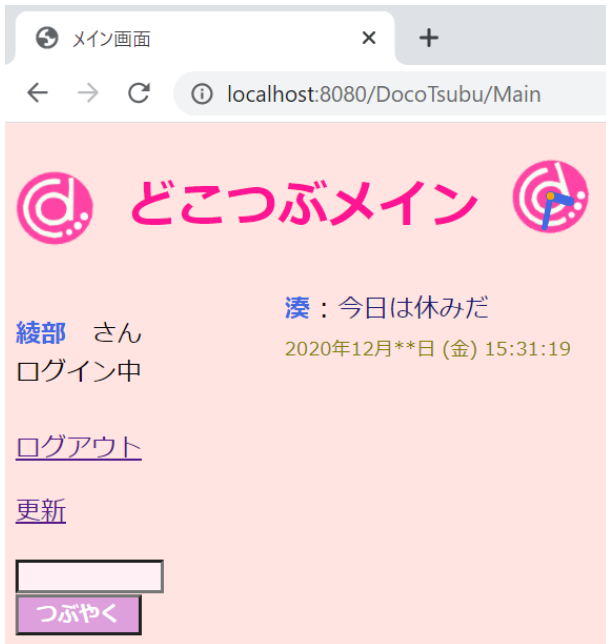
3. つぶやきが表示される

MySQL でデータベースの内容を確認

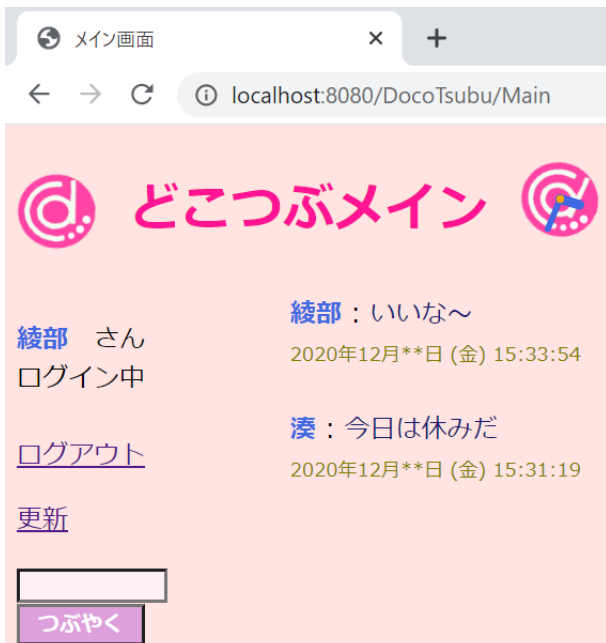
```
SELECT * FROM docotsubu.MUTTER;
```

```
mysql> SELECT * FROM docotsubu.MUTTER;
+----+-----+-----+-----+
| ID | NAME | TEXT          | DATE          |
+----+-----+-----+-----+
| 1  | 湊   | 今日は休みだ  | 2020年12月**日 (金) 15:31:19 |
+----+-----+-----+-----+
1 row in set (0.00 sec)
```

つぶやいた内容がデータベースに登録されている



4. ログアウトし、
ユーザー「綾部」でログイン
5. 別ユーザーのつぶやきも表示されている
6. 「いいな～」とつぶやく



7. つぶやきが追加される

MySQL で確認


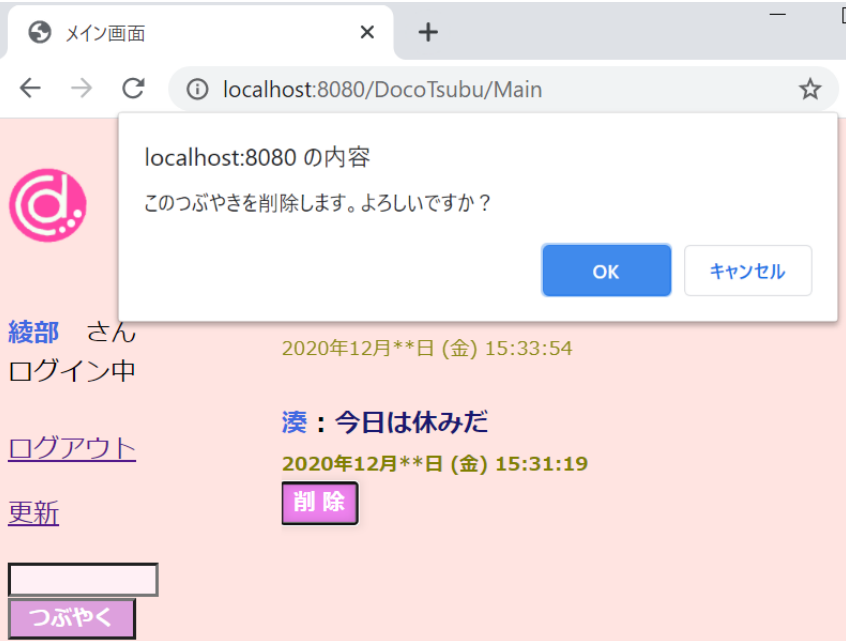

```
SELECT * FROM docotsubu.MUTTER;
```

```
mysql> SELECT * FROM docotsubu.MUTTER;
```

ID	NAME	TEXT	DATE
1	湊	今日は休みだ	2020年12月**日 (金) 15:31:19
2	綾部	いいな～	2020年12月**日 (金) 15:33:54

2 rows in set (0.00 sec)

つぶやいた内容が追加されている

	<p>8. カーソルをつぶやきに合わせると「削除」ボタンが表示される仕様</p> <p>別ユーザーのつぶやきを削除してみる</p>
	<p>9. 確認されるので「OK」</p>
	<p>10. 別ユーザーのつぶやきは削除できない仕様なので、そのように表示される</p>

	<p>11. 自分のつぶやきを削除してみる</p>
	<p>12. 今度は削除成功</p>
	<p>13. 自分のつぶやきが消えている</p>

MySQL で確認

```
SELECT * FROM docotsubu.MUTTER;
```

```
mysql> SELECT * FROM docotsubu.MUTTER;  
+-----+-----+-----+-----+  
| ID | NAME | TEXT          | DATE          |  
+-----+-----+-----+-----+  
| 1 | 湊   | 今日は休みだ   | 2020年12月**日(金) 15:31:19 |  
+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

こちらもデータが削除されている

Creation Date 2020 12