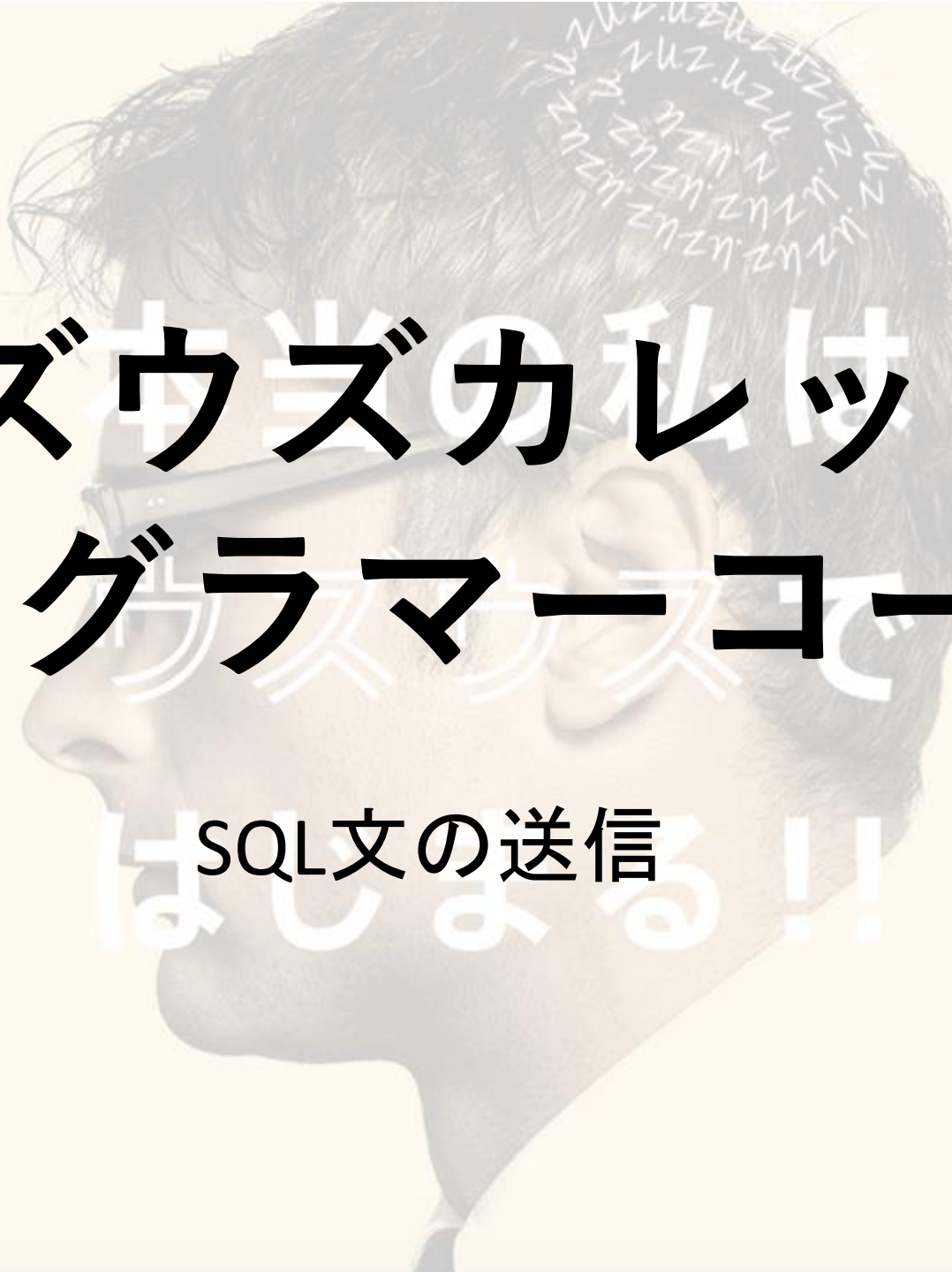


ウズウズカレッジ
プログラマーコース

SQL文の送信

～データベース接続の５ステップ～

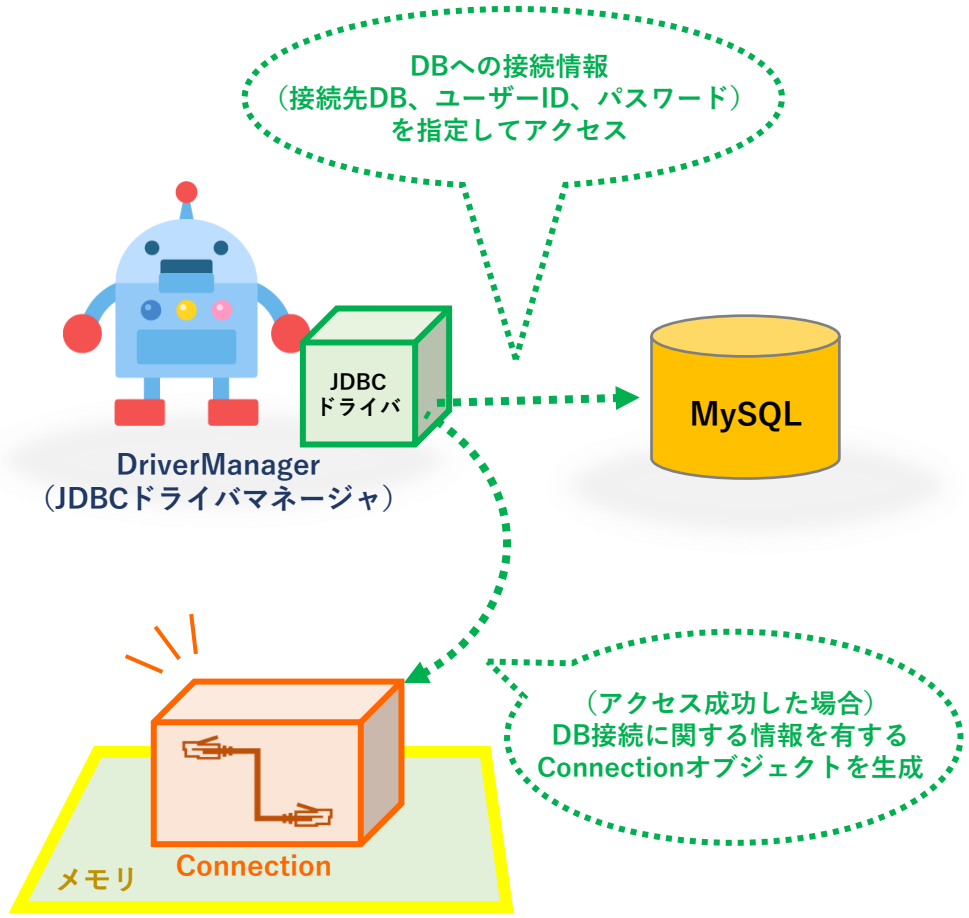
復習

ステップ① JDBCドライバのロード → **ステップ② 接続の確立** → ステップ③ SQL文の送信 → ステップ④ 抽出結果の取得 → ステップ⑤ 接続の解除

```
Connection          con = null ;    // Connection（DB接続情報）格納用変数
```

```
//接続先のデータベース
//※データベース名が「test_db」でない場合は該当の箇所を変更してください
String jdbcUrl    = "jdbc:mysql://localhost/test_db?characterEncoding=UTF-8&serverTimezone=JST&useSSL=false";

//接続するユーザー名
//※ユーザー名が「test_user」でない場合は該当の箇所を変更してください
String userId     = "test_user";

//接続するユーザーのパスワード
//※パスワードが「test_pass」でない場合は該当の箇所を変更してください
String userPass   = "test_pass";
```

```
//-------------------------------------------
// ②接続の確立（Connectionオブジェクトの取得）
//-------------------------------------------
con = DriverManager.getConnection(jdbcUrl, userId, userPass);
```



DriverManagerクラス

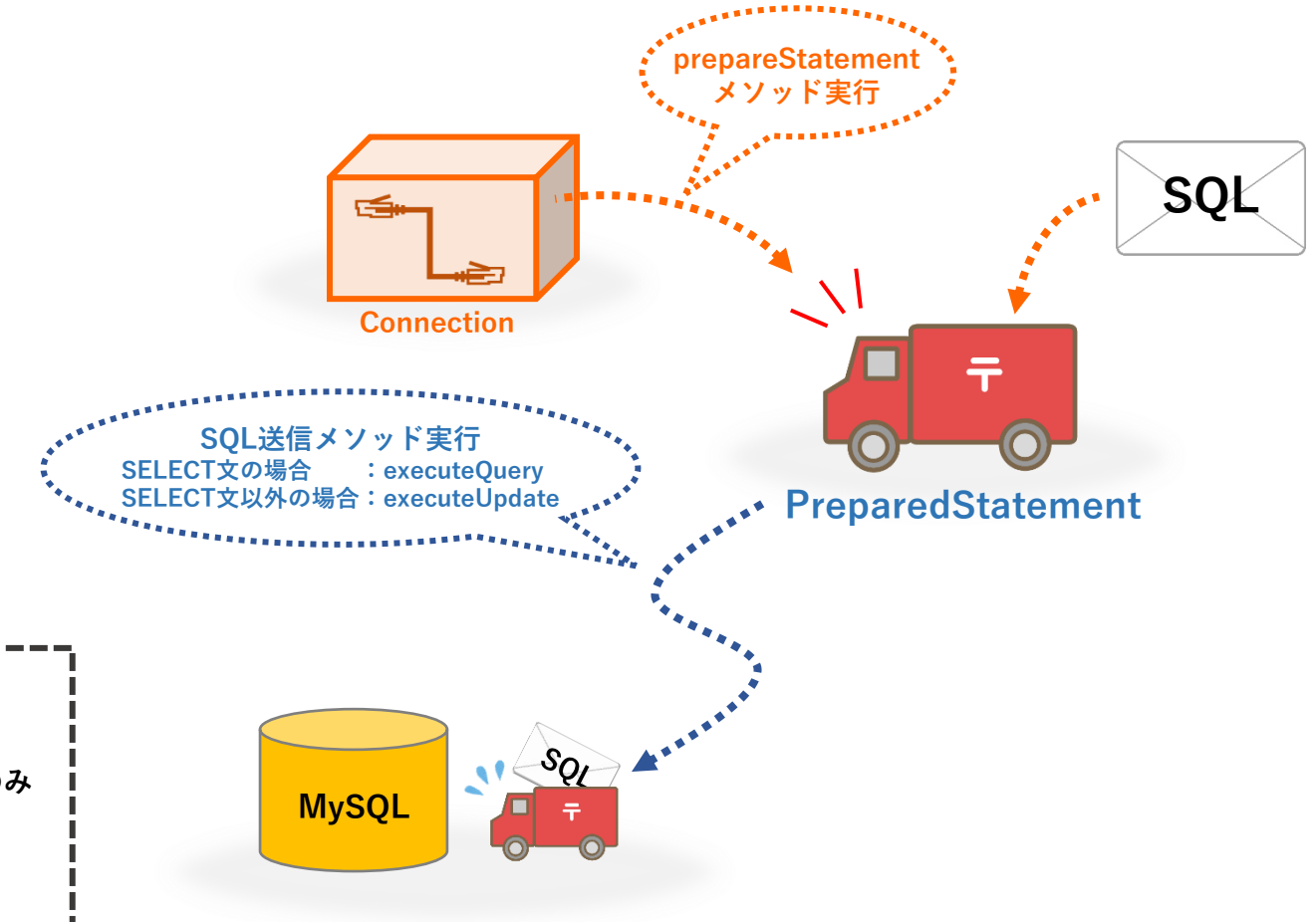
- JDBCドライバマネージャのクラス。JDBCドライバにアクセスしてRDBMSを操作するための様々な機能が提供されている。
- getConnectionメソッドでRDBMSに接続し、接続に成功した場合はDB接続に関する情報を有するConnectionオブジェクトを戻り値として返す。引数にDBへの接続情報（接続先DB、ユーザーID、パスワード）を指定することで接続が実行される。

～データベース接続の５ステップ～

ステップ① JDBCドライバのロード → ステップ② 接続の確立 → **ステップ③ SQL文の送信** → ステップ④ 抽出結果の取得 → ステップ⑤ 接続の解除

```
PreparedStatement ps  = null ;   // PreparedStatement（SQL発行用オブジェクト）格納用変数
```

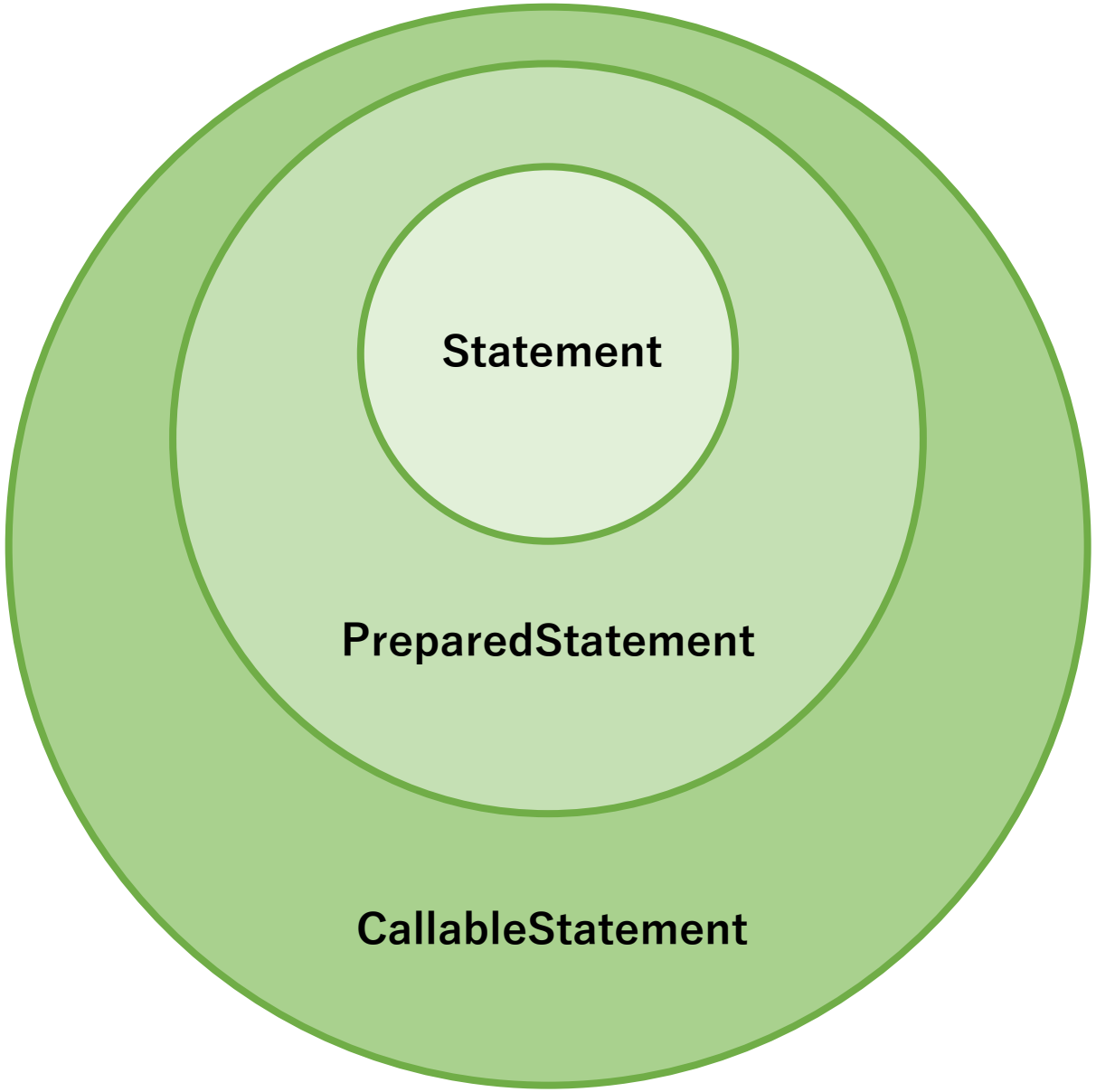
```
//SQL文の生成（SELECT）
StringBuffer buf = new StringBuffer() ;
buf.append(" SELECT         ");
buf.append("    id      ,   ");
buf.append("    name    ,   ");
buf.append("    gender  ,   ");
buf.append("    age     ,   ");
buf.append("    course      ");
buf.append(" FROM           ");
buf.append("    uzuz_member ");
buf.append(" ORDER BY       ");
buf.append("    id          ");

//PreparedStatementオブジェクトを生成＆発行するSQLをセット
ps = con.prepareStatement(buf.toString());

//SQL文の送信＆抽出結果（ResultSetオブジェクト）の取得
rs = ps.executeQuery();
```



Statement系オブジェクト

・RDBMSにSQL文を送信する機能を有するオブジェクト。
　※用途によって複数用意されていますが、**本コースではPreparedStatementにのみ触れます**。
・SQLを送信する際はまず送信したいSQLをセットし、次に送信するためのメソッドを実行する。 このメソッドはSELECT文を送信する場合に用いるexecuteQuery、それ以外に用いるexecuteUpdateの2種類用意されている。

～Statement系オブジェクト～



～データベース接続の５ステップ～

ステップ① JDBCドライバのロード → ステップ② 接続の確立 → ステップ③ SQL文の送信 → ステップ④ 抽出結果の取得 → ステップ⑤ 接続の解除

StringBufferクラス

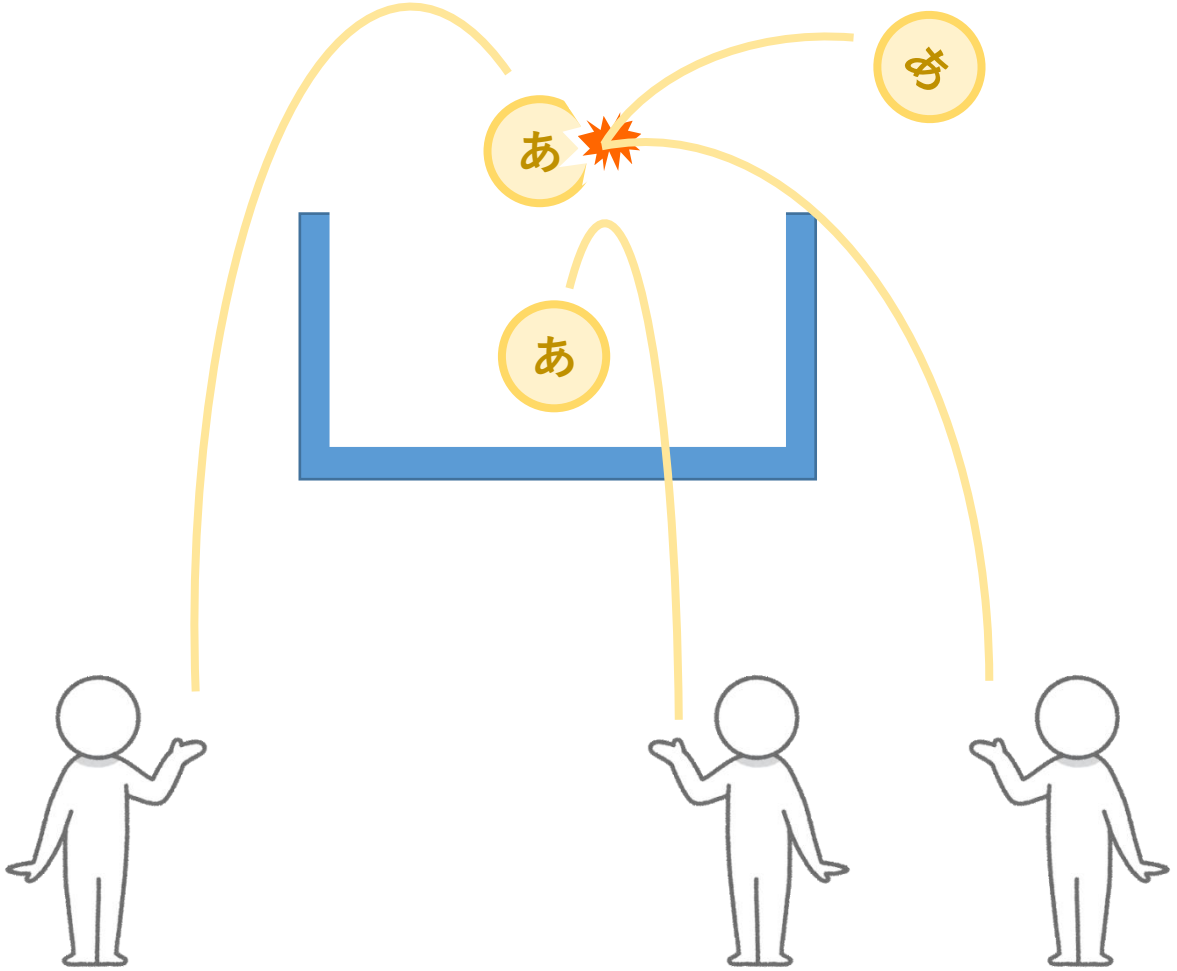
```
//SQL文の生成（SELECT）
StringBuffer buf = new StringBuffer() ;
buf.append(" SELECT         ");
buf.append("   id      ,    ");
buf.append("   name    ,    ");
buf.append("   gender  ,    ");
buf.append("   age     ,    ");
buf.append("   course       ");
buf.append(" FROM           ");
buf.append("   uzuz_member  ");
buf.append(" ORDER BY       ");
buf.append("   id           ");

//PreparedStatementオブジェクトを生成＆発行するSQLをセット
ps = con.prepareStatement(buf.toString());
```

同じ動作

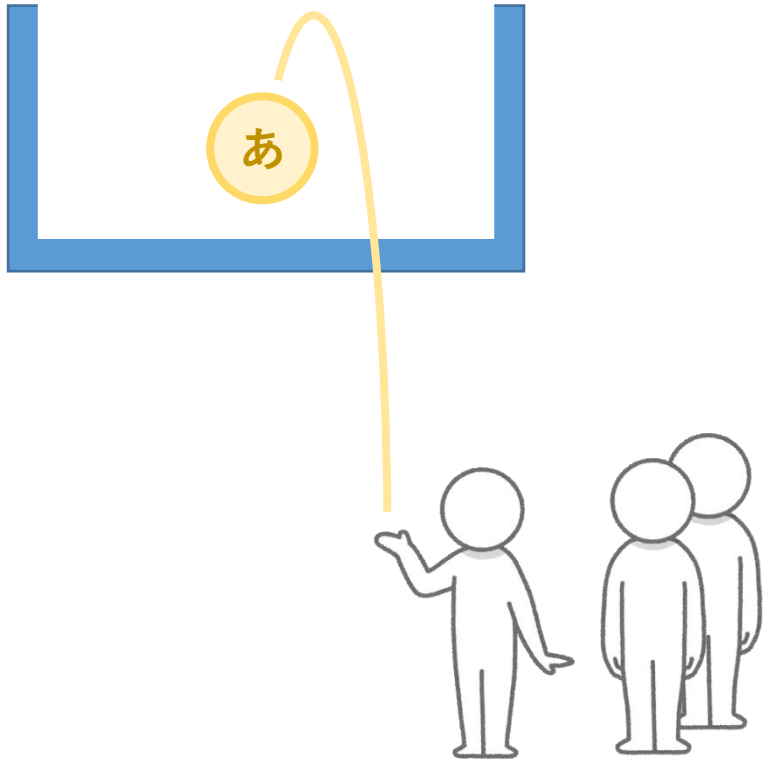
```
//SQL文の生成（SELECT）
String ope = " SELECT         " +
             "   id      ,    " +
             "   name    ,    " +
             "   gender  ,    " +
             "   age     ,    " +
             "   course       " +
             " FROM           " +
             "   uzuz_member  " +
             " ORDER BY       " +
             "   id           "   ;

//PreparedStatementオブジェクトを生成＆発行するSQLをセット
ps = con.prepareStatement( ope );
```

どちらも文字列の結合ですが、結合する文字列が多い場合は
StringBuffer（またはStringBuilder）を使用しましょう。
プログラムの処理スピードに大きな差がでます。

～StringBufferとStringBuilder～



StringBuilder



StringBuffer