JDBC

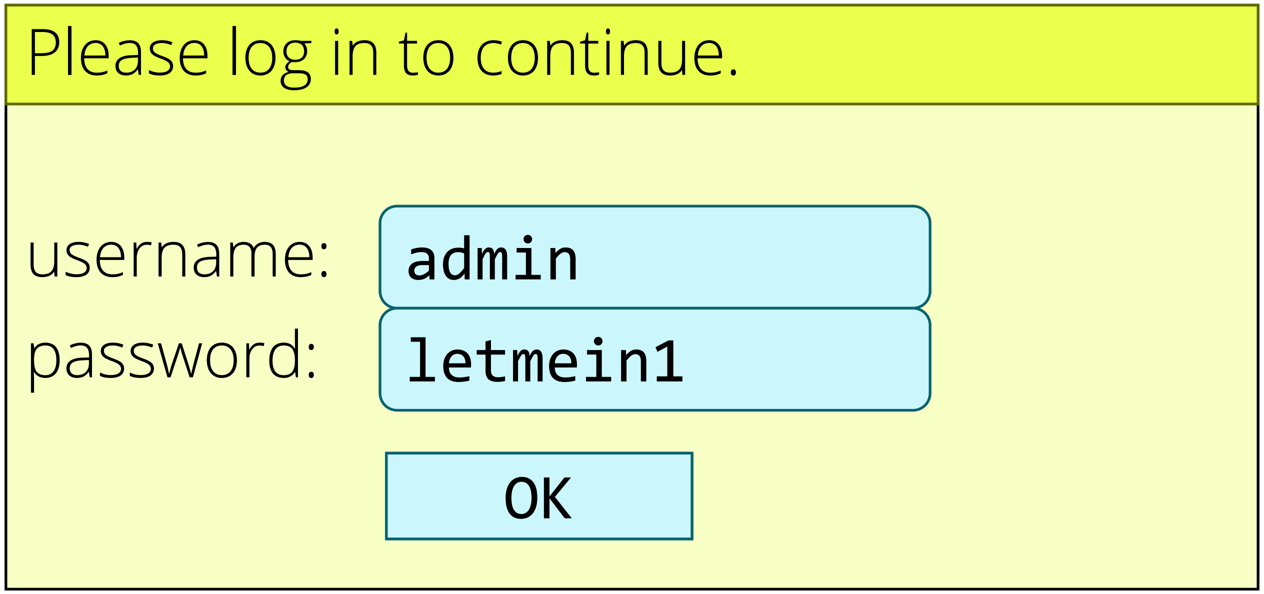
COMS10012 Software Tools

Preparation

Resource

```
try (Resource r = ...) {
    // do stuff with r
}
// when you get here, r.close() is called
// if r is not Null
```

Prepared statements

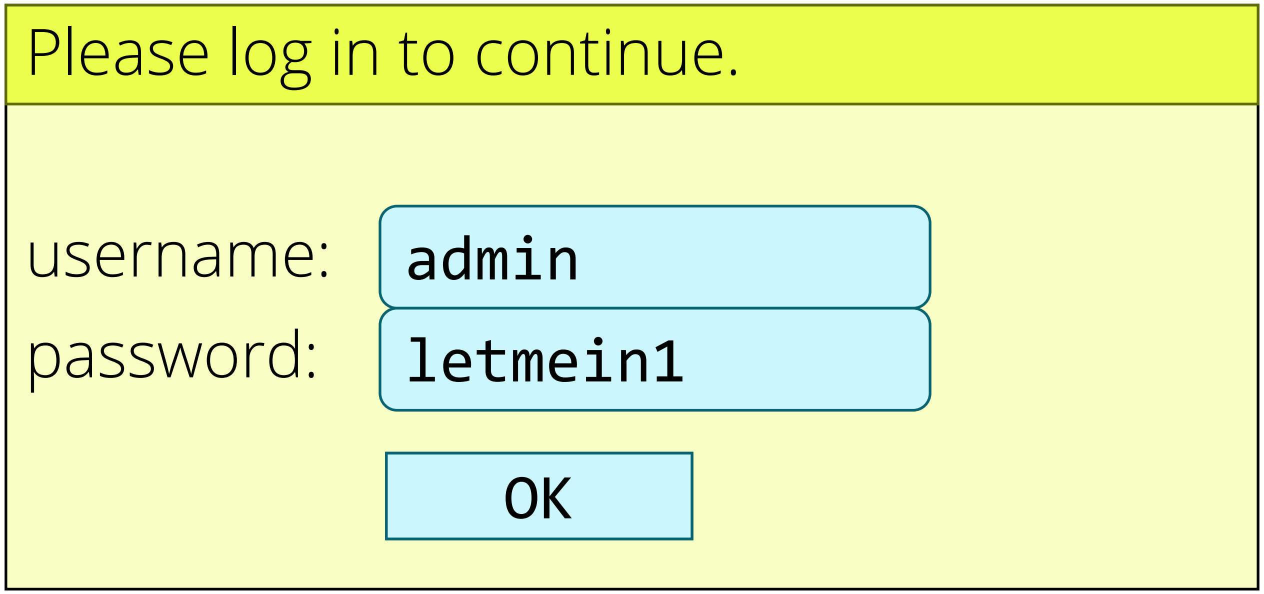


Please log in to continue.

username: admin

password: letmein1

Prepared statements


Please log in to continue.

username: admin

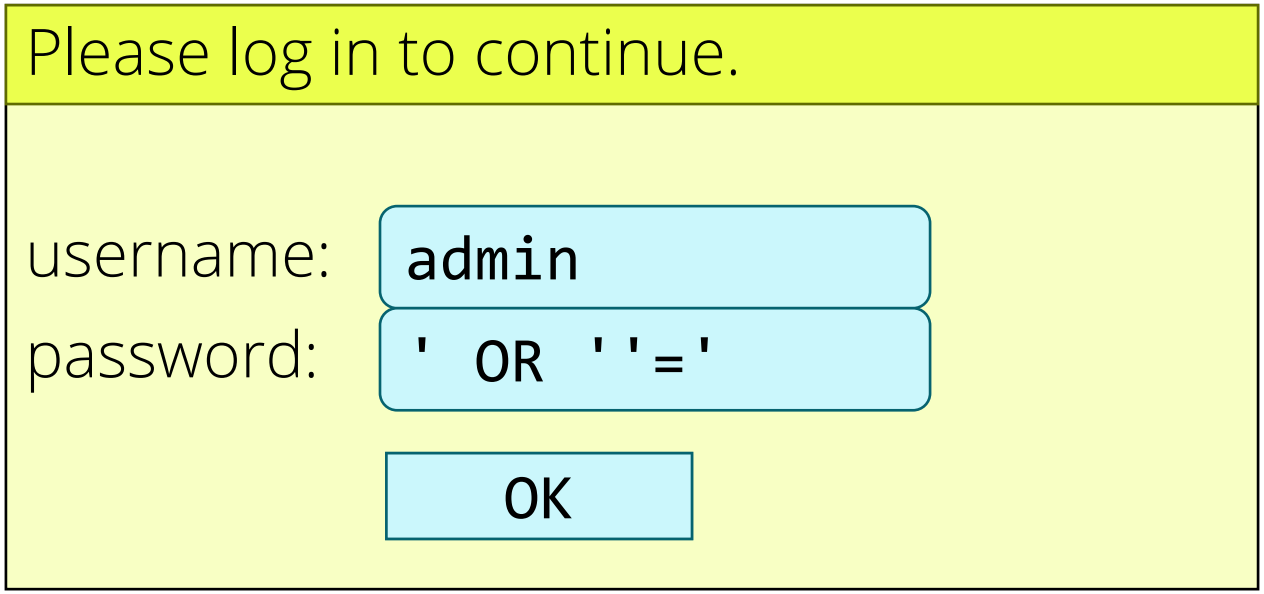
password: letmein1

```
SELECT id FROM Users WHERE
username = 'admin' AND password = 'letmein1';
```

Prepared statements

```
String SQL = "SELECT id FROM Users WHERE " +
  "username = '" + form.field("username") +
  "' AND password = '" +
  form.field("password") + "';";
```

Prepared statements


Please log in to continue.

username: `admin`

password: `' OR ''='`

Prepared statements

```
String SQL = "SELECT id FROM Users WHERE " +
  "username = '" + form.field("username") +
  "' AND password = '" +
  form.field("password") + "';";
```

```
SELECT id FROM Users WHERE
username = 'admin'
AND password = '' OR ''='';
```

Prepared statements

```
s = prepare("SELECT id FROM Users WHERE " +
    "username = ? AND password = ?");

r = s.execute(form.field("username"),
              form.field("password"));
```

JDBC

JDBC

Java DataBase Connectivity

low-level API to interact with databases using a ResultSet abstraction

drivers for all major databases
(but not all of them free)

Connection string

```
public static String connection_string =
"jdbc:mariadb://localhost:3306/elections?
user=vagrant&
localSocket=/var/run/mysqld/mysqld.sock";
```

set up

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

try (Connection c = DriverManager.
    getConnection(connection_string)) {
    // do stuff
} catch (SQLException e) {
    // deal with it
}
```

use

```
try (PreparedStatement p =
     c.prepareStatement(stmt_string)) {
    // bind parameters
    ResultSet r = p.executeQuery();
    // do stuff with results
} catch (SQLException e) {
    // handle it
}
```

parameters

```
p.setInt(1, 100);
p.setString(2, "hello");
```

ResultSet

Reading when there is no row causes a SQLException.

ResultSet starts before the first row.

next() advances a row and returns true if it found one, in which case it's safe to read.

next() returns false when you are out of rows.

ResultSet

```
while (r.next()) {
    // do something with row
    String name = r.getString("name");
    int id = r.getInt("id");
}
```

ResultSet

```
// queries that only ever return one row
if (r.next()) {
    // do something
} else {
    throw new RuntimeException("no row");
}
```