

# PHP and DB

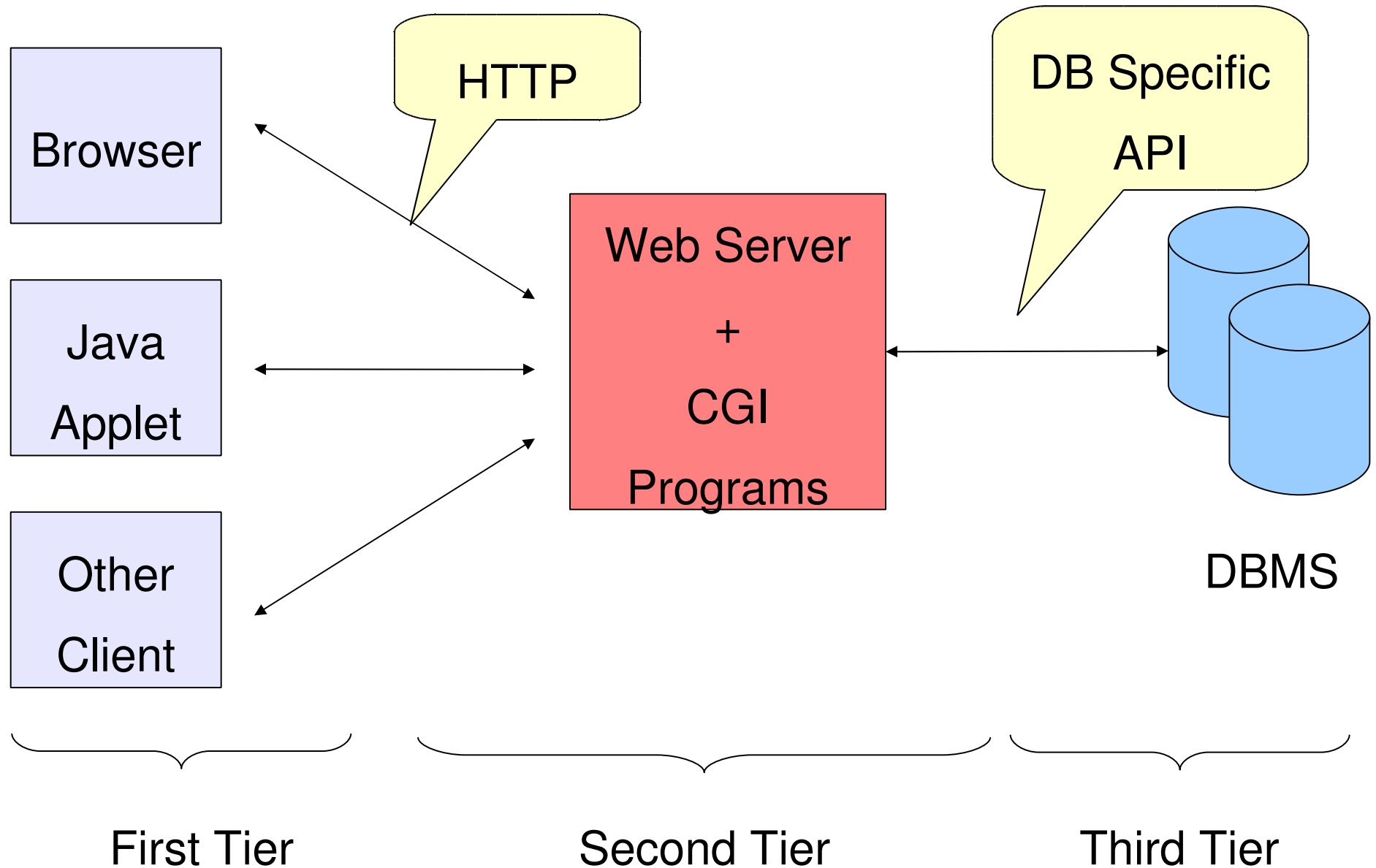
**Massimiliano Pala**

**<massimiliano.pala@polito.it>**

***Politecnico di Torino***

***Dip. Automatica e Informatica***

# Web Database Applications



# First Steps

- **Use the DB extension (/usr/share/pear/DB.php)**
- **To Access a DB you have to:**
  - Generate the Data Source Name (DSN).
  - Connect to the DB server
    - Eventually select the database
  - Build up the SQL query
  - Execute the query
  - Fetch the results
- **Do not forget to check for ERRORS!**

# Identifying the Data Source

```
dbtype://user:password@host:port/database
```

- **dbtype** is one of the supported DBMS (ibase, mssql, oci8, sqlite, dbase, mysql, odbc, storage, fbsql, msql, mysql, pgsql, sybase)
- **user** and **password** are the credentials used to connect to the DB server
- **host** and **port** are the DB server details
- **database** is the DB name

# DSN Example

```
$db_user = 'user';
```

```
$db_pass = 'pwd';
```

```
$db_host = 'localhost';
```

```
$db_name = 'test';
```

```
$db_engine = 'mysql';
```

## # DSN definition

```
$datasource =
```

```
  $db_engine."://". $db_user.":" $db_pa  
  ss."@" $db_host."/". $db_name;
```

# Connecting to a DB server

- to import the DB.php

```
require_once "DB.php"
```

- to connect to the DB by using the DSN

```
$db = DB::connect( $dsn );
```

# Errors, Errors, Errors!

- **A connection to the DBMS may fail because:**
  - the server is not running on the target host
  - the DSN is wrong (hostname, user credentials)
  - the specific DB does not exist
  - It is not possible to create a connection
- **To check for error conditions, use:**

```
if(DB::isError($db))  
    die($db->getMessage())
```

# Executing SQL queries

- the following is used to execute an SQL query:

```
$results = $db->query ( $sql )
```

- where

- **query** is a method of the class DB

- **\$sql** is the variable with the SQL code to execute

- **\$results** is the variable bearing the results

- Return value can be:

- null – an error occurred

- object – query is successful

# Errors, Errors, Errors - Again!

- A query can fail for many different reasons
- You have to check for error conditions, ever!

```
$q = $db->query($sql);  
if(DB::isError($q))  
    die($q->getMessage());
```

# fetchRow()

- When a **SELECT** is performed you have to retrieve data from the **DBMS**

```
$q = $db->fetchRow( $fetchmode, $rownum);
```

- **\$fetchmode:**

- **DB\_FETCHMODE\_ORDERED**

```
$row = array( col1, col2, ..., colN);
```

- **DB\_FETCHMODE\_ASSOC**

```
$row = array( col1_name => col1,  
             col2_name => col2, ..., colN_name => colN );
```

# SQL -1

- **SELECT [ \* | <cols> ] FROM <tables> [ WHERE ... ]**

**SELECT \* from users where name='test1'**

- **INSERT INTO <table> COLS ('col1', ..., 'colN')  
VALUES ('val1',..., 'valN')**

**INSERT INTO users COLS (user,passwd)  
VALUES ('max', 'max')**

- **DELETE FROM <table> [ WHERE ... ]**

**DELETE FROM users WHERE user='max'**

# SQL -2

- **UPDATE SET [ col='val' ... ] [ WHERE ... ]**

**UPDATE SET password='john' where user='max'**

- **In order to work with databases you should**

- Study SQL

- Use STANDARD SQL (portability)

- Carefully design the database

- Normalize the DB

- Create/Destroy indexes for performance issues

# SQL -3

## ■ GRANTING PRIVILEGES

```
GRANT ALL PRIVILEGES ON [db|*].[table|*]  
TO 'max' IDENTIFIED BY  
password('mypasswd')
```

## ■ CREATING A USER

- on MySQL just use the GRANT command
- on Postgresql you have to explicitly create it

```
CREATE USER max WITH PASSWORD 'pwd' ;
```

# SQL -4

- **MYSQL and POSTGRESQL have command line interfaces**

```
mysql -u <username> -p -h <host> <dbname>
```

```
psql -U <username> -W -h <host> <dbname>
```

# db\_wrapper.php - 1

- **Connect()** - connects to the DBMS
- **Connect2()** - connects to the DBMS/databasename
- **Closedb( \$db )** - disconnects from a DBMS
- **Createdb( \$db, \$dbname )** - creates a database
- **Destroydb( \$db, \$dbname )** - drops a database
- **Selectdb ( \$db, \$dbname )** - uses one of the available databases (\$dbname)
- **doQuery ( \$db, \$sql )** - executes an SQL query (\$sql)

# db\_wrapper.php - 2

- **Fetchrow( \$res )** - fetches the next row of data in the dataset (\$res) returned from a doQuery().
  - Returned value is an array
- **FetchrowAS ( \$res )** - same as Fetchrow( \$res ) but the returned value is an associative array ( colname => colvalue, ... )
- **TRACE** - A global constant ( 1 = trace, 0 = quiet )

# PHP and DBMS Resources

- <http://www.php.net>
- <http://www.mysql.com>
- <http://www.postgresql.com>