Advanced Java Application Development Using JavaEE

Day 2

University of Colombo School of Computing

Agenda

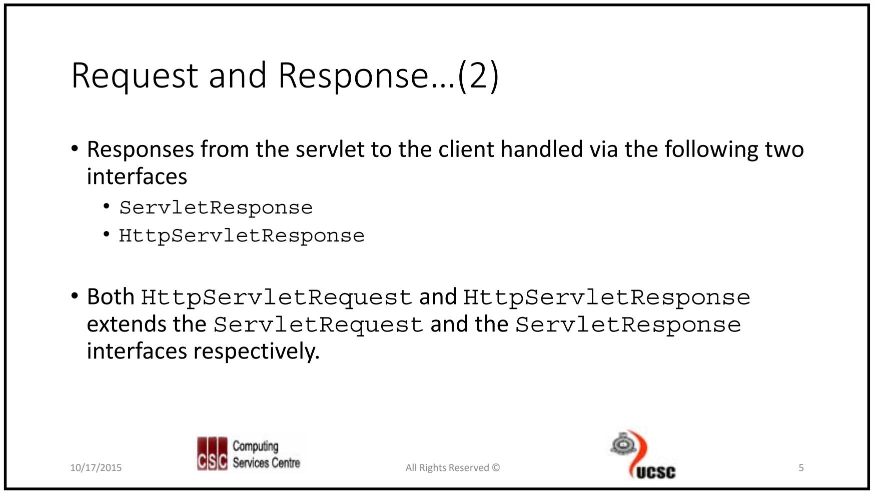
- Preliminaries
- Servlets continued from Day 1
- Development using Eclipse
- Request and Response
- Form Parameters
- Transferring Requests
- Annotations
- JDBC
- ServletConfig and ServletContext
- Attributes
- Sessions

Preliminaries

- Install MySQL Community Server version 5.6.27
  - https://dev.mysql.com/downloads/mysql/
  - Select `Windows (x86, 32-bit) MySQL Installer MSI`
  - Download the web-installer (1.6M)
  - Make sure connector/J is in the downloads list when installing
  - Make sure you remember the passwords you enter
- Download Eclipse
  - Luna or Mars – JEE version

Request and Response…(1)

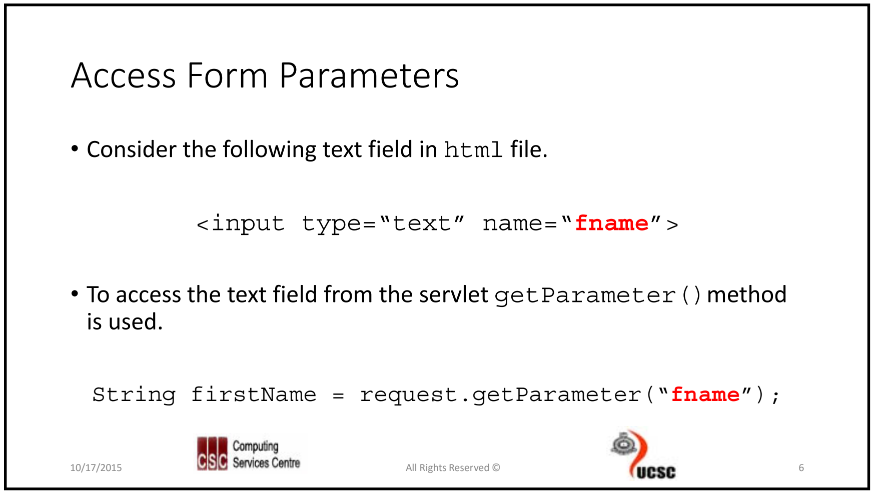
- The purpose of developing servlets is to handle client requests and respond to them.
- Client requests are handled by implementing the `ServletRequest` interface.
- For HTTP specific calls the `HttpServletRequest` interface is used.



Request and Response…(2)

- Responses from the servlet to the client handled via the following two interfaces
  - `ServletResponse`
  - `HttpServletResponse`

- Both `HttpServletRequest` and `HttpServletResponse` extends the `ServletRequest` and the `ServletResponse` interfaces respectively.



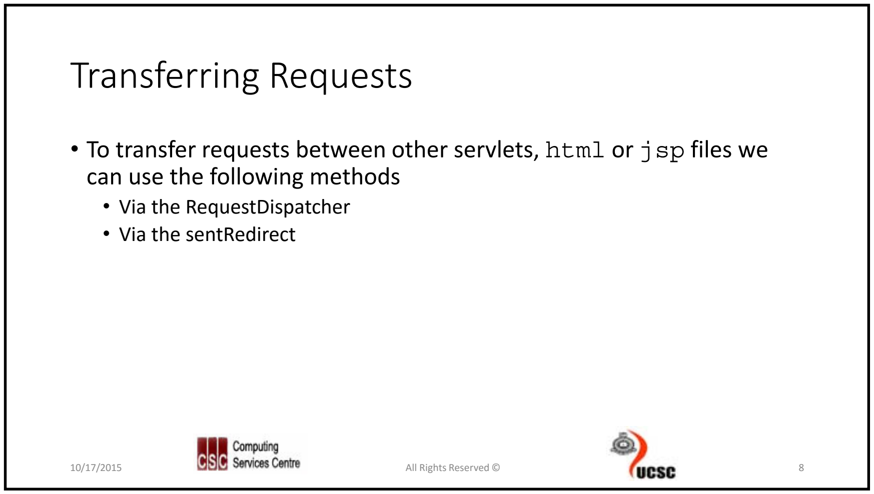
Access Form Parameters

- Consider the following text field in `html` file.

```
<input type=“text” name=“fname”>
```

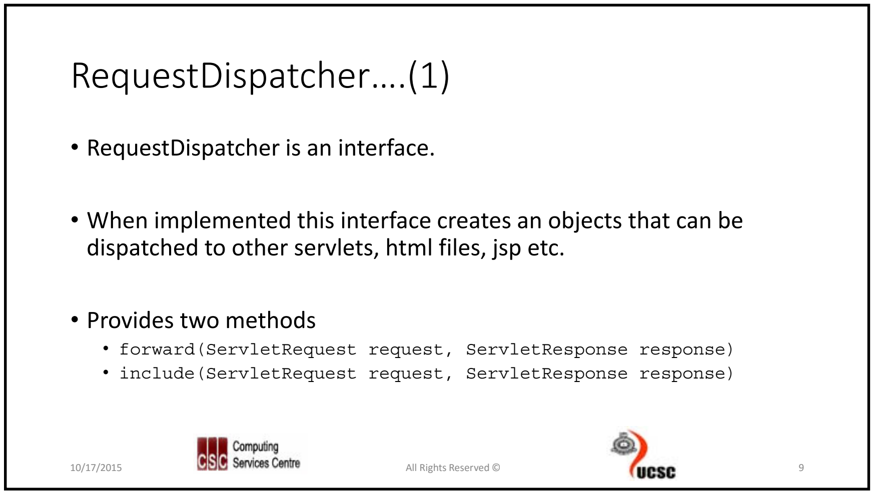
- To access the text field from the servlet `getParameter()` method is used.

```
String firstName = request.getParameter(“fname”);
```



Transferring Requests

- To transfer requests between other servlets, `html` or `jsp` files we can use the following methods
  - Via the RequestDispatcher
  - Via the sentRedirect

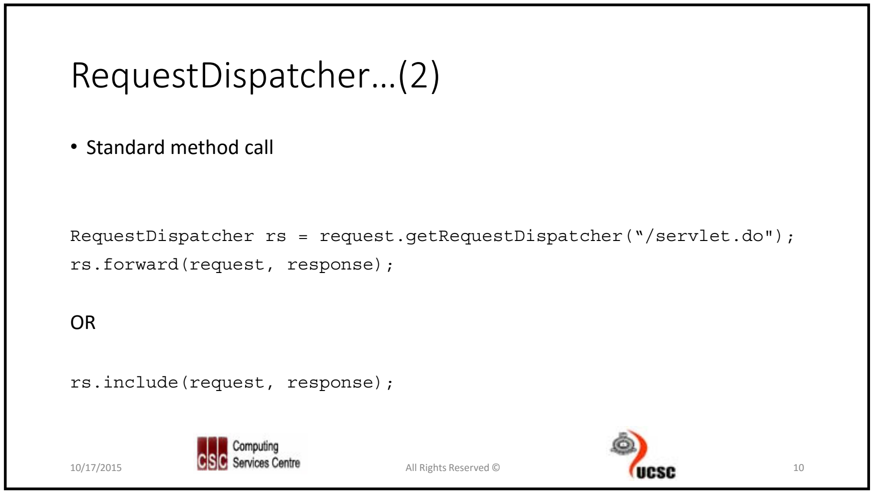


RequestDispatcher….(1)

- RequestDispatcher is an interface.

- When implemented this interface creates an objects that can be dispatched to other servlets, html files, jsp etc.

- Provides two methods
  - `forward(ServletRequest request, ServletResponse response)`
  - `include(ServletRequest request, ServletResponse response)`



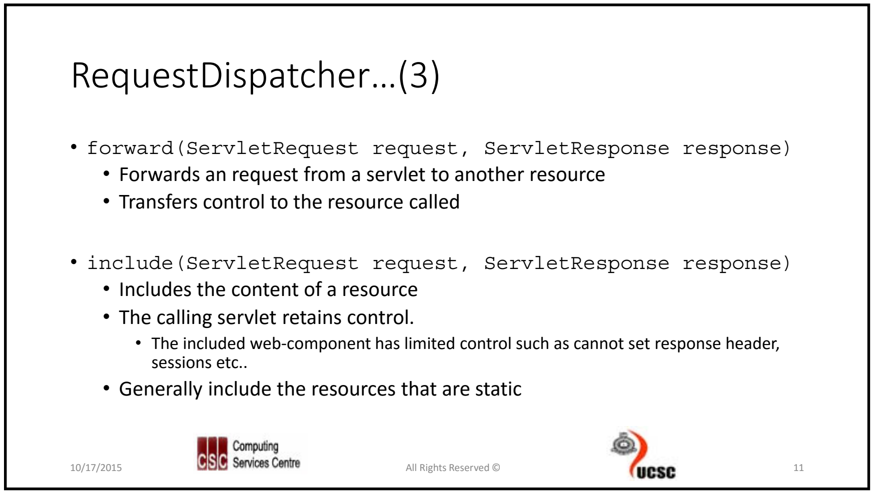
RequestDispatcher…(2)

- Standard method call

```
RequestDispatcher rs = request.getRequestDispatcher(“/servlet.do");
rs.forward(request, response);
```

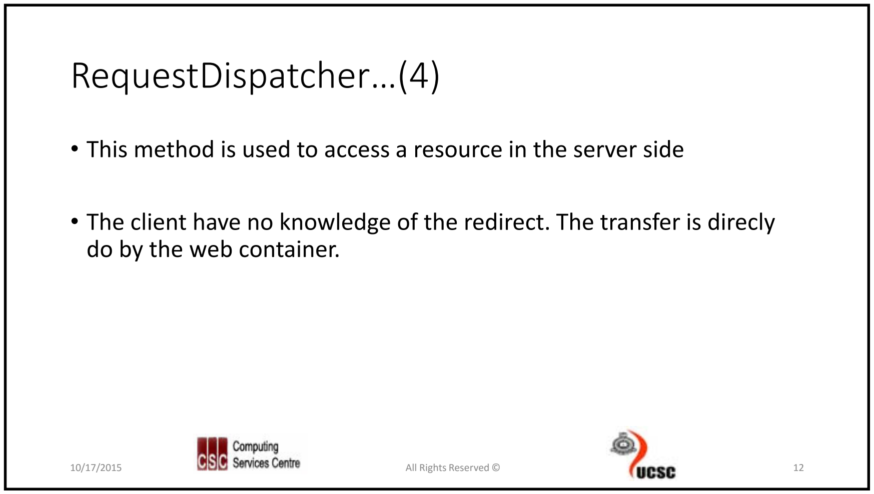
OR

```
rs.include(request, response);
```



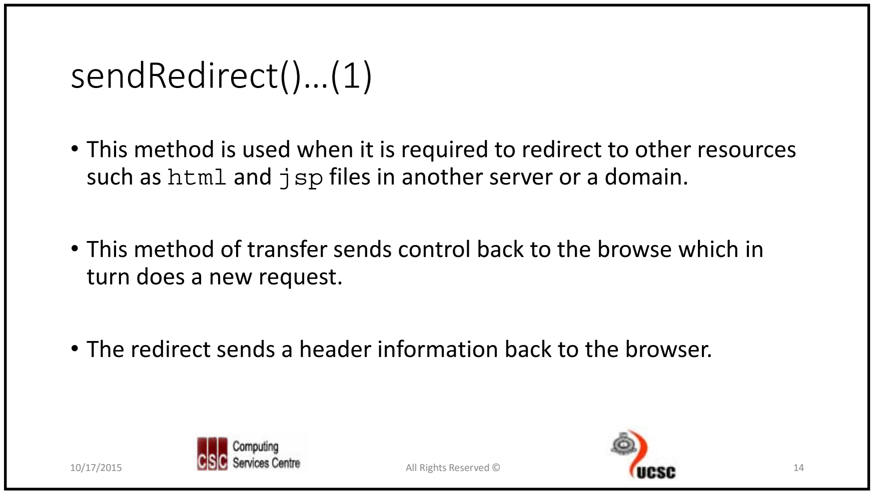
RequestDispatcher…(3)

- `forward(ServletRequest request, ServletResponse response)`
  - Forwards an request from a servlet to another resource
  - Transfers control to the resource called
- `include(ServletRequest request, ServletResponse response)`
  - Includes the content of a resource
  - The calling servlet retains control.
    - The included web-component has limited control such as cannot set response header, sessions etc..
  - Generally include the resources that are static



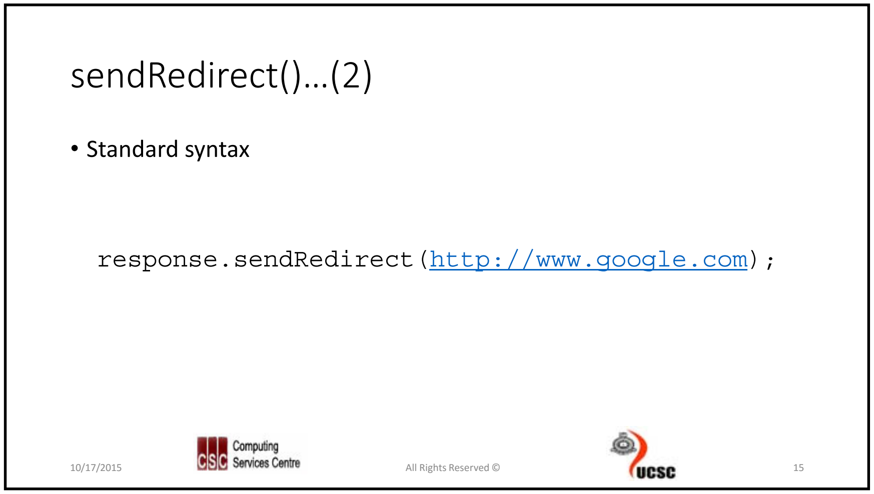
RequestDispatcher…(4)

- This method is used to access a resource in the server side
- The client have no knowledge of the redirect. The transfer is direcly do by the web container.



sendRedirect()…(1)

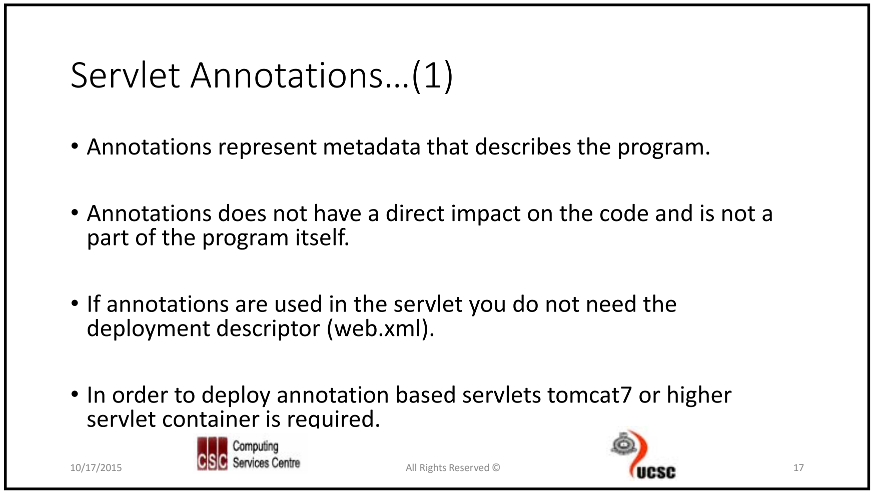
- This method is used when it is required to redirect to other resources such as `html` and `jsp` files in another server or a domain.
- This method of transfer sends control back to the browse which in turn does a new request.
- The redirect sends a header information back to the browser.



sendRedirect()…(2)

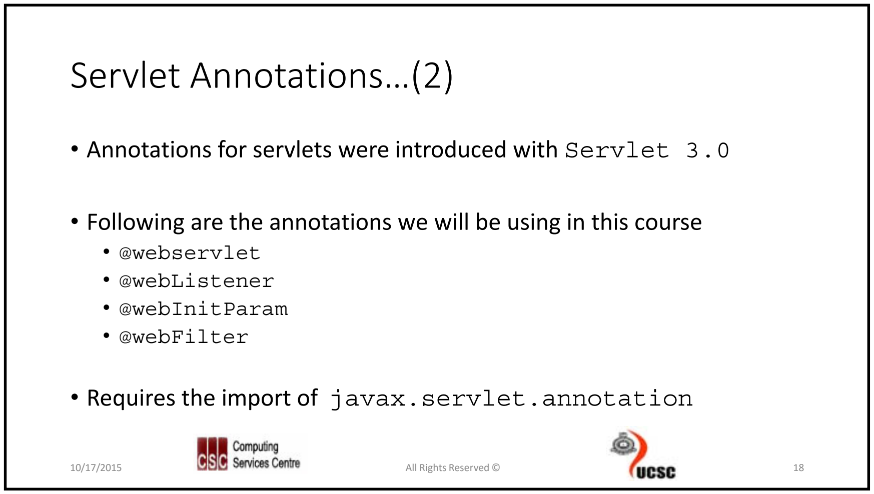
- Standard syntax

```
response.sendRedirect(http://www.google.com);
```



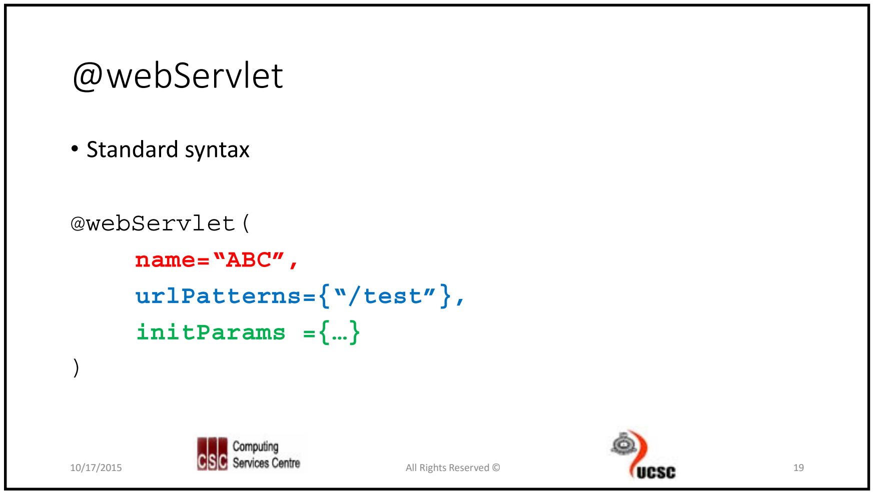
Servlet Annotations…(1)

- Annotations represent metadata that describes the program.
- Annotations does not have a direct impact on the code and is not a part of the program itself.
- If annotations are used in the servlet you do not need the deployment descriptor (web.xml).
- In order to deploy annotation based servlets tomcat7 or higher servlet container is required.



Servlet Annotations…(2)

- Annotations for servlets were introduced with `Servlet 3.0`
- Following are the annotations we will be using in this course
  - `@webservlet`
  - `@webListener`
  - `@webInitParam`
  - `@webFilter`
- Requires the import of `javax.servlet.annotation`



@webServlet

- Standard syntax

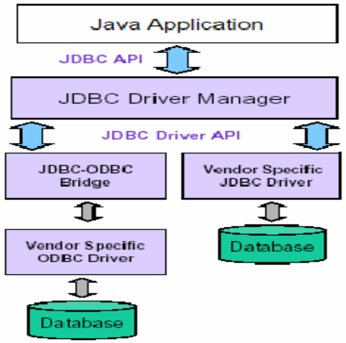
```
@webServlet(
     name="ABC",
     urlPatterns={"/test"},
     initParams ={…}
)
```

Java Database Connectivity (JDBC)

What is JDBC?

- JDBC is an API that lets you access most tabular data sources from the Java programming language
- practice JDBC enables you to read from relational databases, spreadsheets etc…

General Architecture



Java Application

JDBC API

JDBC Driver Manager

JDBC Driver API

JDBC-ODBC Bridge

Vendor Specific JDBC Driver

Vendor Specific ODBC Driver

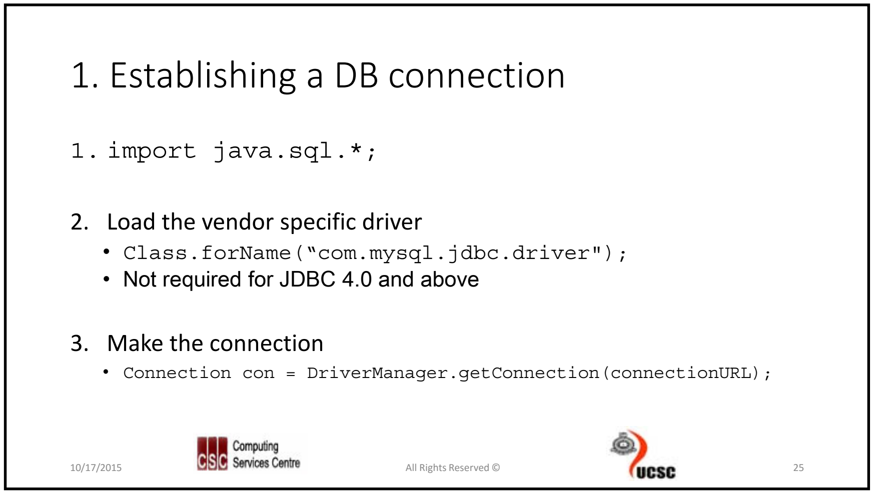
Database

Database

Basic steps to use a database in Java

Any database interaction with JDBC adheres to the following steps.

1. Establish a connection with the database
2. Create JDBC statements
3. Execute SQL queries and operations.
4. Get the ResultSet
5. Close connections

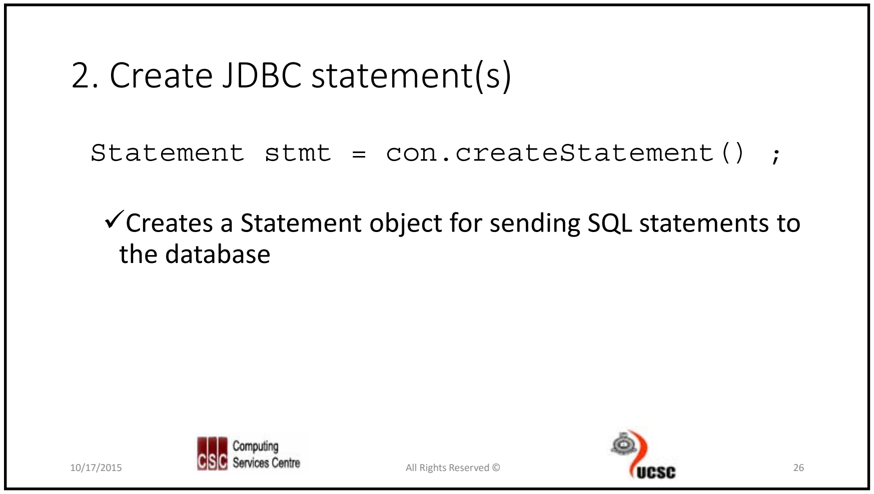


1. Establishing a DB connection

```
1. import java.sql.*;
```

2. Load the vendor specific driver
   - `Class.forName(“com.mysql.jdbc.driver");`
   - Not required for JDBC 4.0 and above

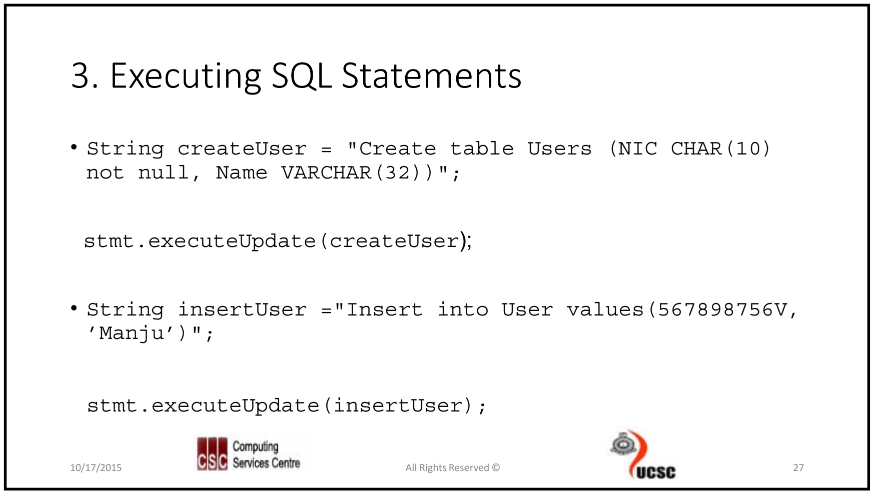
3. Make the connection
   - `Connection con = DriverManager.getConnection(connectionURL);`



2. Create JDBC statement(s)

```
Statement stmt = con.createStatement() ;
```

✓Creates a Statement object for sending SQL statements to the database



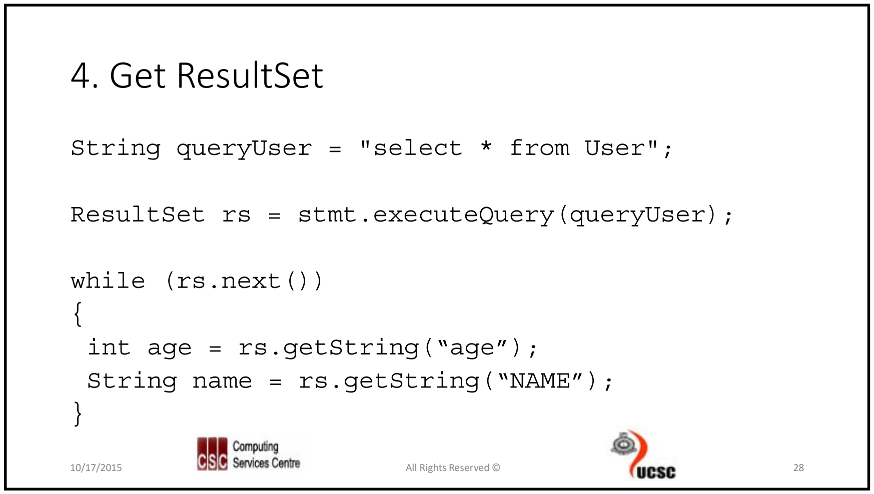
3. Executing SQL Statements

- `String createUser = "Create table Users (NIC CHAR(10) not null, Name VARCHAR(32))";`

```
stmt.executeUpdate(createUser);
```

- `String insertUser ="Insert into User values(567898756V, ’Manju’)";`

```
stmt.executeUpdate(insertUser);
```



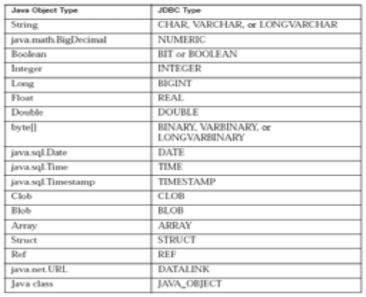
4. Get ResultSet

```
String queryUser = "select * from User";

ResultSet rs = stmt.executeQuery(queryUser);

while (rs.next())
{
 int age = rs.getString(“age”);
 String name = rs.getString(“NAME”);
}
```

Mapping Types



Java Object Type	JDBC Type
String	CHAR, VARCHAR, or LONGVARCHAR
java.math.BigDecimal	NUMERIC
Boolean	BIT or BOOLEAN
Integer	INTEGER
Long	BIGINT
Float	REAL
Double	DOUBLE
byte[]	BINARY, VARBINARY, or LONGVARBINARY
java.sql.Date	DATE
java.sql.Time	TIME
java.sql.Timestamp	TIMESTAMP
Clob	CLOB
Blob	BLOB
Array	ARRAY
Struct	STRUCT
Ref	REF
java.net.URL	DATALINK
Java class	JAVA_OBJECT

ServletConfig and ServletContext…(1)

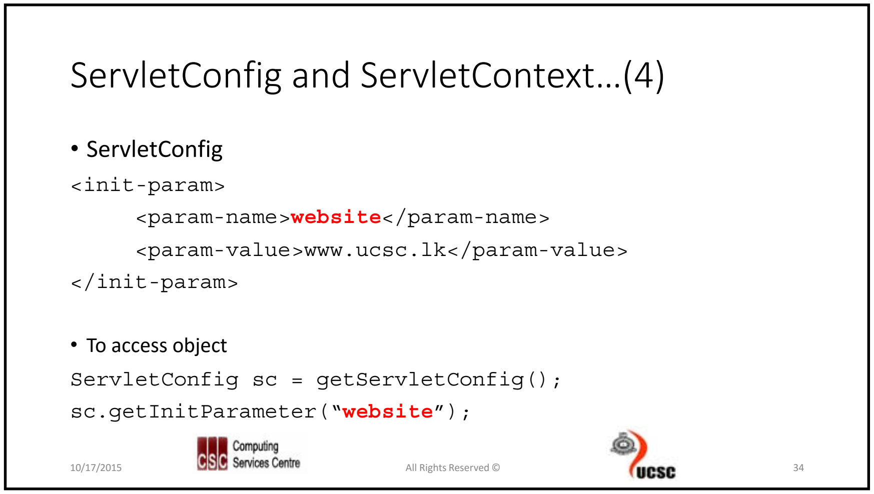
- Both `ServletConfig` and `ServletContext` are interfaces.
- Both the interfaces allows you to define configurations that are used by the servlet container to initialize and pass various parameters of the web application and specific servlets.
- `ServletConfig` object is created for a specific servlet at the initialization of the servlet.

ServletConfig and ServletContext…(2)

- `ServletContext` object has an global effect and is defined at the web application level.
- `ServletContext` object used to obtain information from the web.xml and the information is available to any servlet or JSP that is a part of the web application.

ServletConfig and ServletContext…(3)

ServletConfig	ServletContext
Applies to a single servlet	Applies to the entire web application
Resembles a local parameter associated with that particular servlet	Resembles a global parameter
Defined inside the `<servlet></servlet>` tags of the `web.xml`	Defined as a direct child of the `<web-app></web-app>` tags of the `web.xml`
`getServletConfig()` method retrieves the object	`getServletContext()` method retrieves the object
Used for servlet specific information	Used for application specific information



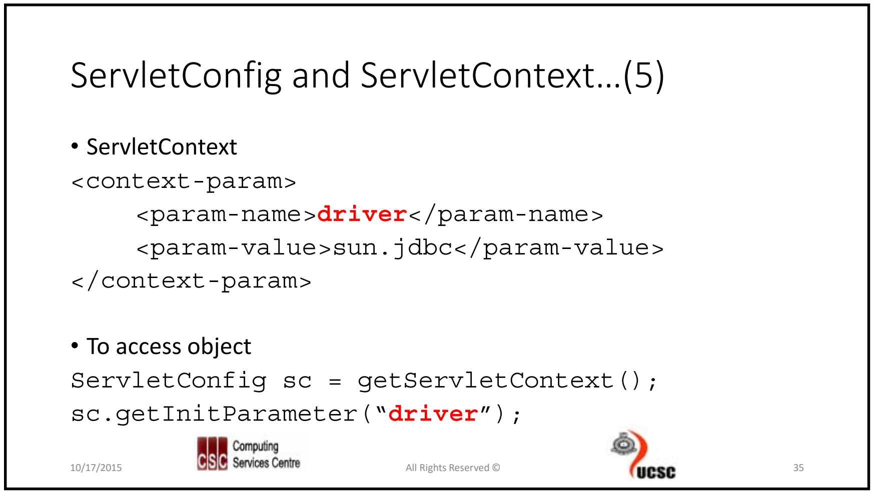
ServletConfig and ServletContext…(4)

- ServletConfig

```
<init-param>
     <param-name>website</param-name>
     <param-value>www.ucsc.lk</param-value>
</init-param>
```

- To access object

```
ServletConfig sc = getServletConfig();
sc.getInitParameter("website");
```



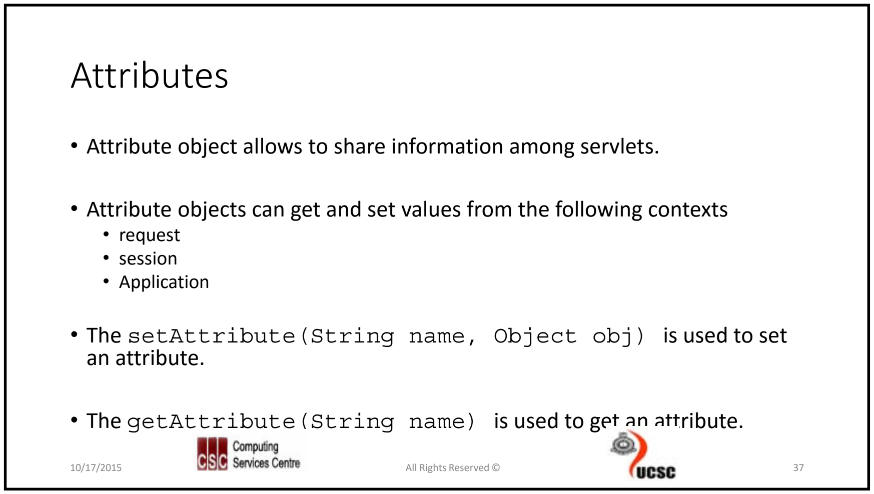
ServletConfig and ServletContext…(5)

- ServletContext

```
<context-param>
     <param-name>driver</param-name>
     <param-value>sun.jdbc</param-value>
</context-param>
```

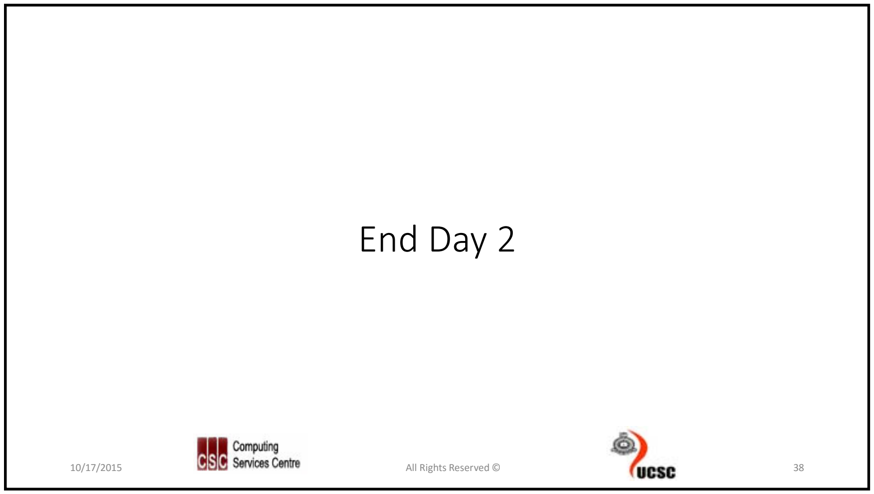
- To access object

```
ServletConfig sc = getServletContext();
sc.getInitParameter("driver");
```



Attributes

- Attribute object allows to share information among servlets.
- Attribute objects can get and set values from the following contexts
  - request
  - session
  - Application
- The `setAttribute(String name, Object obj)` is used to set an attribute.
- The `getAttribute(String name)` is used to get an attribute.



End Day 2