

INT202

COURSE DESCRIPTIONS

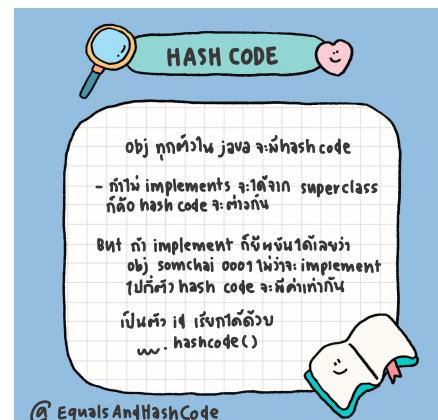
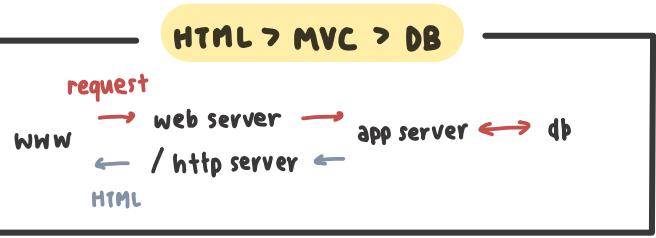
- Web (World Wide Web) architectures
 - HTML (Hypertext Markup Language)
 - HTTP (Hypertext Transfer Protocol)
 - request, responses, request parameters, header

- Web application architectures

- Application servers
- Simple server-side scripting
- MVC (Model view controller) Design Pattern
- Application state management: cookies + session management (ดูใช้แบบบุ๊ฟตอนไหนแล้ว = state)
- ORM (Object-Relational Mapping) 1 object 1 ตารางบันทึก
- Creating, testing, deploying simple web applications
- Standard tools, frameworks, technologies for web

DEPENDENCIES

- library ตัวช่วยในการเขียนgetter setter
- เก็บ dependencies ไว้ใน pom.xml
 - เวลาหาเจ้าของ local (\Users\Spewwii\.m2\)
 - But ก้านไม่เจอก็ไปหานอกจาก central repositories (internet)
<http://repo1.maven.org/maven2/>



JCF (JAVA COLLECTION FRAMEWORK)

- Array (Hash)
 - พื้นที่จด - ขนาดคงที่
 - ลับเลือดสร่าย้ำบ้าด้วย
 - โครงสร้างมีลักษณะเดียวกัน
 - ไม่สามารถแก้ไขการเปลี่ยนแปลง
- Linked (Tree)
 - ไม่มีพื้นที่จด
 - รับบุคคลบ้าด้วย
 - โครงสร้างมีลักษณะแบบผังวงกลม
 - สามารถแก้ไขการเปลี่ยนแปลง

ABSTRACT DATA STRUCTURE

- list > ArrayList, LinkedList
- Set > TreeSet (BST), HashSet
- Map > TreeMap, HashMap
- Queue, Stack > linkedlist

hash - obj นั่นก็ต้อง implement hashCode และ ต้องต่างจาก hashCode ไปเก็บ (ห้ามใช้ส่วนตัว = ห้ามต่อหนัง)
 ปรับปรุงการบันทึก load factor ดูจากขนาด table กันจนเข้มข้น
 (1.0 ก็ต้องตั้งปอนด์ = ขนาด table)



sort ใช้ comparable

comparable เป็น接口ที่มี sort ตั้งแต่กัน

tree - obj ที่มี comparable เท่านั้น
 เพราะว่าการใส่ต้น ออกจาก การเรียง
 เพราะวันนี้เราลงต้องเมื่อตัวที่เทียบเป็นไป
 (บล๊อกใจขนาด !)

HASH!

```
Set<Integer> set = new HashSet<> (16, 0.75f);
    ↓
    ↓ loadfactor
```

- จากต้น ที่เราระบุที่จะหดตัว 16 ช่อง But
 สามารถได้เพิ่ง 12 เพราะว่าตัว loadfactor อยู่ภายใต้ range use
 แต่ 75% เท่านั้น มีอยู่ก่อนการใช้บันทึกกับเจ้าตัวช่วงซ่า

TREE!

ก้าวต่อ TreeSet และ

```
set.add(new Student(01,"Som",3.56));
    ↑ error เพราะเทียบตัวบ้าด้วย
    ✗ เพราะว่า obj student ยังไม่ implement
        comparable
```

INT202-01-2566-JCF-Reviews.pdf

Problem 2

Map [key , Value]

ຈາກ array ຂອງເລີງຈຳນວນເຕີມໄດ້, ທີ່ມີຄຳໄໝ້ກັນ ຈະເຫັນໃປໂປຣແກຣມ ເພື່ອແສດງຄູ່ລຳດັບຂອງຂ່ອງ ທີ່ນີ້ກັນ
ແສ້ວ ໄດ້ຄຳ ເປັນ X ເນື້ອ X ອີ່ຄຳໄດ້, ທີ່ຜູ້ໃຊ້ກຳນົດ ໄດຍໃຫ້ໂປຣແກຣມມີ Running Time ເປັນ $O(n)$

0	1	2	3	4	5	6	7	8
1	2	3	5	8	7	9	6	4

ເນື້ອ X = 6

ຈະໄດ້ຜົດລັບນີ້ (3, 0) (8, 1)

(1.0 ກໍດັກເກີ່ມ ພົມວຸ = ນາດ table)

(ບໍ່ພິຈາລະນາ !)

HASH!

excel

www

ล แล้วอยู่ com Ex email upload download

1. Clients = browsers [ส่งไปด้วย HTTP Requests] ที่ server ต้อง url
2. Server = Web Server [ตอบกลับด้วย HTTP Responses]

HTTP Protocol

- 1) Client ใช้ url เพื่อไป server $\text{http://host:port/path/file} \rightarrow \text{ip address (domain name)}$
- 2) Browser send "request" message เป็น text protocol
- 3) Server แปลงเอกสารนั้นๆ
- 4) Server send "response" message เป็น text protocol
- 5) Browser formats + display

Web Application * จำเป็นต้องมี Application Server

- ต้องมี Application Server เก็บ รับ request + ให้ response (แทน web server)
- ล ต้องบันทึก HTML ไว้ส่งกลับไปใน web server

Application Server

- มี 2 container
- 1) Web container (เริ่มนั้น) ดูแลร่อง JSP เขียน HTML ทำงาน data ของ servlet มาและ
 - servlet 1 ต่อ / flow (ก่อต่อๆ กัน)
 - รับ request + เขียน HTML บุฟ มาก : (
 - 2) EJB Container

* glass fish เป็น Java app server

Servlet = เป็น standard ! เขียนเหมือนกัน deploy ไหนก็ได้ เป็น java class ทำงานบน jakarta EE ทำงานบน HTTP request

- ↓ - จะมีการส่ง parameter 2 ต่อ 1) HttpServletRequest request → ใจ text มาต่อกัน obj ใจ header + data
รับจาก url

ทำงานในตัว server

- manage by
web container

ล อาจมี url mapping

```
package com.ibm.example.servlet;
import javax.servlet.http.HttpServlet;
...
public class VerySimpleServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response) throws ServletException, IOException {
        String browser = request.getHeader("User-Agent");
        response.setStatus(HttpServletResponse.SC_OK);
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<HTML><HEAD><TITLE>Simple servlet</TITLE></HEAD><BODY>");
        out.println("<TITLE><HEAD><BODY>");
        out.println("Browser details: " + browser);
        out.println("<BODY><HTML>");
    }
}
```

flow - ต้อง servlet จะเป็นส่วนหนึ่งของ url (mapping) ไม่ใช่ชื่อ class

- Web server → App Server → Web container → ไปนา ก้าวต่อไป: ห่อต่อ new

L เรียก method ชื่อ service ก้าวแบบ get → doGet | ไฟล์รันแรก: เขียน

- forward data ให้ jsp เพื่อส่งกัน ! post → doPost | method in 2 ตัวนี้

MVC

- servlet เป็นตัวรับมา → ส่งข้อมูลให้ jsp มาแสดงผลลัพธ์

ล ไปดึง model มาทำงาน → ตัวจาก db

Benefits

1. ใจ Model นำไปใช้ได้กับหลายที่
2. คลายความซับซ้อนของ App But ต้องเพิ่มคนมาทำ (model, view, controller)
3. ใจ view, model, controller ได้แบบไม่วางผูกขาดกัน

Controller + View

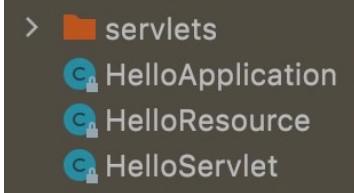
SERVLET + JSP

JSP!

`<h1></. = "Hello world" </.>`
↓
บันทึกว่าข้างในเป็น java

การเรียกไฟล์ servlet นี้ใช้

``
↓
การท่องเว็บดังต่อไปนี้เป็นตัวอย่าง url บน.



↑ เวลาเรียกใช้อ่านจาก value

```
@WebServlet(name = "helloServlet", value = "/hello-servlet")
public class HelloServlet extends HttpServlet {
    private String message;
```

in servlet นี้ method หลักๆ 2 อย่าง 1) doget()
2) dopost()

```
public void init() { message = "Hello World!"; }

public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
    response.setContentType("text/html");

    // Hello
    PrintWriter out = response.getWriter();
    out.println("<html><body>");
    out.println("<h1>" + message + "</h1>");
    out.println("</body></html>");
}
```

JAKARTA SERVER PAGE (JSP)

↑ response คือ

content
└ static ผูก logo / Header ไม่พึ่งการเปลี่ยนแปลง
dynamic ฟังก์ชันทางโปรแกรม
ทำให้เรา mix static+dynamic ได้
- ไม่พึ่ง JSP syntax, HTML

= JSP คือ ในการเขียนข้อมูลจากท่อนุญาตให้เราเขียน HTML กับ Scripting เพื่อให้เราเขียนให้ดีกว่าให้ทำ้งานในฝั่ง server และผู้รับดูจะเห็นเหมือนอ่านจดหมาย

รีบต้องรู้ ดู ใจดี

JSP SYNTAX

- Directives
- Scripting (Declaration, Expressions, Scriptlets)
- Actions
- ~ Comments

VARIABLES (ตัวแปรที่)

- request: HttpServletRequest Object
- response: HttpServletResponse Object

JSP SCRIPTING

1. Scriptlet (พิเศษ Java ล้วนๆ)

```
<% valid_code_fragment %>
    ทำให้เขียนภาษา java ห่างไกล
<% if (Calendar.getInstance().get(Calendar.AM_PM) == Calendar.AM) %>
    How are u this morning?
<% else %> How are u this afternoon? <% %>
```

2. Expressions (สามารถใช้กับชนิดใด = ตัวที่มันไม่ผลลัพธ์ของ)

`<% = expression %>`

Ex calls incrementCounter method ประจำวัน

`<% = incrementCounter() * 3 + y %>`

```

package sit.int202.simpleweb.servlets;

import ...

@WebServlet(
    name = "StudentListServlet",
    value = {"student-list"}
)
public class StudentListServlet extends HttpServlet {
    public StudentListServlet() {
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        StudentRepository studentRepository = new StudentRepository();
        * request.setAttribute("students", studentRepository.all()); // ป้อน collection ให้ JSP ด้วย obj
        request.getRequestDispatcher("/StudentList.jsp").forward(request, response);
    }
}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
}

```

servlet

```

<%          ต่องบอกด้วยว่าต้องการ 가져 collection ที่มีอยู่ในไฟล์ JSP นี้
Collection<Student> students = (Collection<Student>) request.getAttribute("students");
for (Student student : students) {%
    <div class="col-2">           นำ java file student
        <div>Id: <%= student.getId()%></div>
        <div>Name: <%= student.getName()%></div>
        <div>gpax: <%= student.getGpax()%></div>
        <div><hr></div>
</div>

```

JSP

ตัวจัดไฟล์ servlet มาก

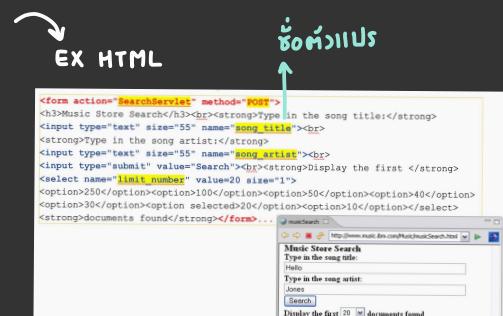
HTTP PROTOCOL : request

request phase

- Request (POST)
- Header values
- (เก็บบรรทัด)
- Posted data

```
POST /Music/SearchServlet HTTP/1.1
Accept: */
Referer: http://www.music.ibm.com/Music/musicSearch.html
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
UA-CPU: x86
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; ...)
Host: localhost:9080
Content-Length: 50
Connection: Keep-Alive
Cache-Control: no-cache
song_title=Hello&song_artist=Jones&limit_number=20
```

↓
ชื่อตัวแปร
↓
request parameter



Servlet

การจัดบ้าน Request (เวลาดูมุมมอง)

- getParameter (String name) : ลักษณะเป็น String
- getParameterMap () : ส่วนรับ multiple (ในการนี้ได้รับ value)
- getParameterName () : ลักษณะเป็น enum + เอกสารได้ชี้
- getParameterValue (String name) : return value ก็จะง่าย

Ex $x=50 \& y=30 \& name=Somchai \& subject=Jakarta \& subject=database$

`request.getParameter ("name") = Somchai`

`request.getParameter ("x") = 50`

`request.getParameter ("Subject") = Jakarta`

↳ ถ้าต้องการทราบวันที่ที่เราต้องการค่าตามห้องสมุดควรใช้แบบล่วงๆ

`request.getParameterValue ("subject") = { Jakarta, database }`

`request.getParameterName () = enum { "x", "y", "name", "subject" }`

HTTP PROTOCOL: response

response phase

- status information
- Header values
- (เก็บบรรทัด)
- output document

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=ISO-8859-1
Content-Language: en-US
Set-Cookie: JSESSIONID=000093gEM_2OosI_mR6GBkZLJy9:-1; Path=/
Transfer-Encoding: chunked
Date: Mon, 12 Mar 2007 18:32:27 GMT
Server: WebSphere Application Server/6.1
Expires: Thu, 01 Dec 1994 16:00:00 GMT
Cache-Control: no-cache="set-cookie, set-cookie2"
<HTML>
<BODY>
<H1>Very simple dynamic document created on 01-Jun-2001</H1>
</BODY>
</HTML>
```

status information

1xx information

2xx Success

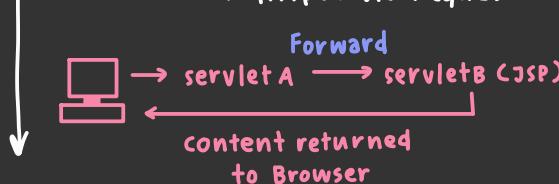
3xx Redirection (ไปหน้าจอเพิ่มขึ้น)

4xx Client Error (request ไม่ถูกต้อง)

5xx Server Error (code พัง)

Request Dispatcher

- ค้นหา resource ที่อยู่ในโปรเจกต์
- แบ่งจาก 2 ที่ 1) ServletContext
2) HttpServletRequest



- use

- 1) Forward to jsp page

ส่งไป servlet A

`getServletContext().getRequestDispatcher("/pages/webBalance.jsp").forward(request,response);`

- 2) include static HTML

`getServletContext().getRequestDispatcher("/pages/navigation-bar.html").include(request,response);`

SHARING OBJECT

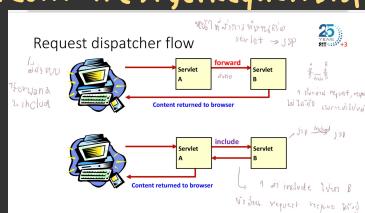
- ServletContext

- 1) `getServletContext().setAttribute ("objname", anObj);`
- 2) `getServletContext().getAttribute ("Objname");`

- HttpServletRequest

- 1) `request.setAttribute ("Objname", anObj);`
- 2) `request.getAttribute ("Objname");`

↳ หมายเหตุ: get ต้องผูกกับ set มาก่อนต้องจะใช้กับ servlet



- วิธีการทำงานของ servlet คือ

```

<form action="add-new-student" method="post">
    ID: <input type="number" name="id" required><br>
    Name: <input type="text" name="name" required><br>
    Gpax: <input type="number" name="gpax" max="4.0" required><hr>
    <input type="submit" value="Save">
</form>

```

JSP
FORM

```

package sit.int202.simpleweb.servlets;

import ...

@WebServlet(
    name = "AddNewStudentServlet",
    value = {"/add-new-student"}
)
public class AddNewStudentServlet extends HttpServlet {
    public AddNewStudentServlet() {
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        this.getServletContext().getRequestDispatcher(s: "/StudentForm.jsp").forward(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String id = request.getParameter(s: "id");
        String name = request.getParameter(s: "name");
        String gpax = request.getParameter(s: "gpax");
        Student student = new Student(Integer.valueOf(id), name, Double.valueOf(gpax));
        file java
        StudentRepository studentRepository = new StudentRepository();
        studentRepository.save(student);  ↗ ชื่อส่วนต่อไปนี้เป็นตัวแทน object ก็จะบันทึกใน page นี้
        request.setAttribute(s: "newStudent", student);  ↗ กับไปบันทึก !
        this.getServletContext().getRequestDispatcher(s: "/NewStudentInfo.jsp").forward(request, response);
    }
}

```

JSP EL §§

↗ ถ้า obj java มาอยู่ใน jsp

JSP expression Language (EL) = simple language

- ถ้าต้องไปส์ java มาได้

- ใช้แค่กันก็ได้

- ผู้ relational, logical, arithmetic

- ถ้า obj ที่อยู่ใน web container จะ (implicit)

- ทำ function ไว้จะเละไปเรียก

- ใช้ร่วมกับ HTML tag , JSTL

↑ meaning = มันเรียกพ่อ method บ้าได้รีบกันผ่าน attribute
แปลว่าต้องมี getter, setter หรือต้องซื้อความธรรมเนียมเพื่อให้เรียกใช้ได้

* ถ้าเกิดไม่พบ ↑

<h2> Hello, \${user.firstname} \${user.lastname} </h2>

* เป็นกัน custom tag (JSTL)

<c:if test = "\${tag3}"> ... </c:if>

IMPLICIT OBJECTS

- | | |
|--------------------|---------------|
| - pageContext | - param |
| - pageScope | - paramValue |
| - requestScope | - headerValue |
| - sessionScope | - cookie |
| - applicationScope | - initParam |

Ex <!-- request.getAttribute("name") -->
~~~~~ ↓  
\${requestScope.name}

## SYNTAX OVERVIEW 🐝

- [] and .  
used to Maps, Lists, Arrays of obj
- Arithmetic, logical, Relational comparisons
- access ต้องไปรอดำเนินด์ ผ่านชั้นตัวแปร attribute ที่อยู่ใน obj ขาดไป

## BASIC SYNTAX ELEMENTS 🦷

- Literals : Boolean, int, float, string, null
- Operator : Accessor (. [ ] ), Arithmetic (+ - \* / . .), Relational ( ==, !=, <, >, <=, >= ),  
Logical ( & &, ||, ! ), Empty (empty ), conditional ( ? : )

Ex \${param.cost \* 1.085}

~~~~~ \${empty sessionScope ? "yes" : "no"}

EX

```
package sit.int202.simpleweb.models;

public class Student {
    private Integer id;
    private String name;
    private Double gpax;

    public Integer getId() { return this.id; }

    public String getName() { return this.name; }

    public Double getGpax() { return this.gpax; }

    public void setId(Integer id) { this.id = id; }

    public void setName(String name) { this.name = name; }

    public void setGpax(Double gpax) { this.gpax = gpax; }

    public Student(Integer id, String name, Double gpax) {
        this.id = id;
        this.name = name;
        this.gpax = gpax;
    }
}
```

MODEL

<h2>New Student has been added</h2>

Student Id: \${newStudent.id} เรียกจาก getter
Student Name: \${newStudent.name}

Gpax: \${newStudent.gpax}

JSP1



* การเรียกใช้ในได้รีบกัน attribute
พันเรียกจาก method getter

* จุดเด่นของการใช้ param และ ตัวแปร

```
<input type="hidden" name="x" value="100">
<input type="hidden" name="x" value="900">
<input type="hidden" name="x" value="400">
```

request.param.id: \${param.id}

request.param.name: \${param.name}
 param = parameter
request.param.gpax: \${param.gpax}

x = { \${paramValues.x[0]}, \${paramValues.x[1]}, \${paramValues.x[2]} }

x = {
 <c:forEach items="\${paramValues.x}" var="v" varStatus="vs">
 \${vs.count}: \${v} ผล!!
 </c:forEach> x header. User-Agent X
}

 ที่ต้องที่หน้ามี - กรณีของ: ชื่อ. บล็อก
User-Agent = \${header["User-Agent"]}] ต้องมี [] และ

JSP2



JSTL (library von custom tag)

- ໃນໂນໂລບໍ່ທີ່ສ່ວງເພື່ອເປັນນັບນັກ JSP
- custom tag = ກາຮສ່ວງ tag HTML ຫັນພາກຕ່າງໆ
- └ ໂປດໄຈ້ XML (ໃຫດນັບນັກ java ທ່ານັດນັກ view)
- └ ຈະເກີດ JSTL ທີ່ເປັນນາມຕ່າງໆ!
- ຜິ 4 ຊົດ
- * 1) Flow (iteration + conditionals)
- 2) Manipulation of XML document
- * 3) Internationalization tags
- 4) SQL tags

SAMPLE JSTL TAGS

- set

ກະບຸ scope ວ່າໃນຍຸແນ
↑ ມັດຕະເປັນວິຊ
`<c: set var = "name" scope = "scope" value = "expression" />`

- out ແສດງພລວັດ

`<c: out value = "expr" default = "expr" escapeXml = "boolean" />`
Ex Hello `<c:out value = "${user.name}" default = "Guest" />`

- conditional (if else)

`<c:choose>, <c:when>, <c:otherwise>`

Ex `<c:choose>`

`<c:when test = "${user.role == 'member'}">`
`<p> welcome, member! </p>`
 `</c:when>` // ພັນການ case ອີ when ຢາງອັນ!
 `<c:otherwise>`

`<p> welcome, guest ! </p>`
`</c:otherwise>`

`</c:choose>`

`<c:forEach items = "${students}" var = "s" varStatus = "vs">`
`<div class = "col-2 p-1 m-2 border border-secondary">`
`${vs.count%5==1 || vs.count%5==3 ? 'bg-primary' : ''}>`

ເປັນກາຮແກ່ຈາກນອາເກົາ
ນາບ el + jstl ແກ່ນ!

- forEach ຮູ່ນ collections, Maps, Iterators, Enum, Array

Ex `<table>` ເປັນພາກ collection, array

`<c:forEach items = "${customer}" var = "cust">`
`<tr><td> ${cust.name} </td>`
 `<td> ${cust.addr} </td>`
 `</tr>`
`</c:forEach>`

`</table>`

↑ ໂຄນຈາກ request in StudentListServlet
`<c:forEach items = "${students}" var = "s">` ກຳນົດໄລຍ້ຕ່າງໆໃນ loop ເປັນ s
 `<div class = "col-2 p-1 m-2 border border-secondary">`
 `<div> Student Id: ${s.id} </div>`
 `EL`
 `<div> Name: ${s.name} </div>`
 `<div> Gpax: ${s.gpax} </div>`
 `</div>`
`</c:forEach>`

`x = {` EL `ແລ້ວໄຫວ່າຕົວ (v: value)`
`<c:forEach items = "${paramValues.x}" var = "v" varStatus = "vs">`
 `${vs.count}: ${v} >` ຖອນດີດູບ!
`</c:forEach>`
} `
`

↓ ເກັບບົດຂຶ້ນຂອງກາຮນູນ
ຝ 1) index ຮອນກີ່ 0, 1, 2, 3
2) COUNT ລົດລວມ!

ເຖິງ ຄົນ ຕົ້ນ
`<c:forEach begin = "1" end = "100" var = "value">`
 `<div class = "box"> ${value} </div>`
`</c:forEach>` ຮູ່ 1 - 100

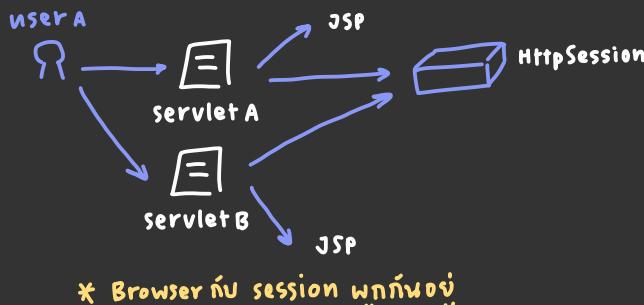
ງານນາ!

HTTP SESSION (obj ທີ່ເກີບ session)

- ກໍາໄລຈະເກີບຂອງລາຍ request → forward → response ແລ້ວຂອງລາຍ request ຈະໄສສາມາດໃຫ້ວາໄດ້ປັດ (ຕ້ອງບໍ່ດີ)
- Obj ທີ່ເຮັດວຽກກີບໃຫ້ user ໂຕລະຄົມແລ້ວຕົວໆການໃນເຖິງ request ກົນໄດ້ຕົວໆໃຊ້ Session ຂອງລາຍ: 0ບຸ້ຈະກ່າວ
- Method `request.getSession(boolean create)`
 - ↳ ກໍາເກີດຢັງໄວ້ເຊັ່ນພື້ນນະ: ສ້າງ session ໃນພື້ນນະ
 - But ກໍາພື້ນ session ອຸ່ນ ແລ້ວກົດໃຫ້ຕົວເຄີນ
- ການໄຫວຂອງລາຍຈາກເກີບ `void setAttribute(String key, Object value)`
 - ↳ servlet → object
 - But ກໍາເມີນ JSP ກໍາລື່ງ retrieve

```
request.getSession()
(true) = ກໍາມີຂອງລາຍເຕັມນາໄຟ
↓
ມີມານີ້ແລ້ວນີ້ຈະສ້າງ session ຖ້າແລ້ວ
(FALSE) = ກໍາມີຂອງລາຍເຕັມນາໃຫ້
↓
ມີມານີ້ຈະເປີນ null ໄຟສ້າງໃນນີ້!
* () ຕັ້ງໃຈ parameter = (true)
```

timeout
ເບີນໂດດທ່ານໃຫຍ່ໄປກ່າງ
ປັດ browser



- * ກໍານີ້ user B ມາເຮັດ servlet A, B ຈະໄປເປົ້າໃນ HttpSession ວິວຕົວໄວ້ (ໄວຣະວັນບັ້ງລາຍໃຫ້ເຫັນບັ້ງກັນເພື່ອທີ່)
- * ກໍາພົດພເວົ້າໄຟ 10,000 ອຸ່ນ ຜົກ໌ session
 - ບັນບຸ້ງກົບຈຸນ, user ທີ່ຜູ້ໃຈວານໃຈມານີ້ນີ້

```
@WebServlet(
    name = "CourseRegisterHistoryServlet",
    value = "/CourseRegisterHistory")
public class CourseRegisterHistoryServlet extends HttpServlet {
    public CourseRegisterHistoryServlet() {
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        HttpSession session = request.getSession( b: false); // ກອນສ່າງເບີນການໃຫ້ False ແບບເຊົາດນວກເບີນນາເກືອງຊີ່ງກຳຈະຕົ້ງ session ມາປັດ
        if (session == null || session.getAttribute( s: "courseRegistered") == null) { // But ກໍານີ້ໃນໄວ້ຈຳເປົ້າໃຫ້ session ໄຟພະຍາຍດີມີຂ່າຍ
            request.setAttribute( s: "message", o: "ໄມ້ມີຂໍອ້ມູດ ກາລົງທະເບີນ ຂອງທີ່ລັງທະເບີນກອນ");
            // ຢັ້ງໃຫ້ show ເປັນນັອງ jsp
            this.getServletContext().getRequestDispatcher( s: "/ShowRegisterHistory.jsp").forward(request, response);
        }
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }
}
```

ລົກປະການຂອງ EL ໂຈາຕ້ອງຕົວແປງຈະດີ

- 1) PageScope
- 2) RequestScope
- 3) SessionScope
- 4) ApplicationScope

ຕໍ່ໄຟ້ນີ້ກັບ 4 ຕົວກີ່ນີ້ທີ່ຈຳກັດໄດ້

- ຊື່ອຕົວໄປປາເສັບກົນກໍາໃຫ້ຕ້ອງ: ບຸ້ scope ເຮັດວຽກ!

↓

ສະນຸ້ມຕົວໄເກີດຕົວແປງ X ໃນກົດ 4 scope

X ທີ່ໄວ້ໄດ້ຈະມາຈາກ PageScope!

ກໍາຮັບບຸ້ scope ກົດ: ຮູ່ວ່າໄຮຈະ: ເຕົາ X ຂອງ scope 1 ນັ້ນ

ໃປທ່າ method doPost

✓ voic ↑

ການໃຊ້ EL+JSTL in JSP

```

<select name="semester" id="semester" class="px-4">
    <c:forEach items="#${semesters}" var="semester" varStatus="vs">
        <c:if test="#{semester != null}"> ໂຕີມາເລັດຈໍາດ້ານ semester ເມື່ອ null ສັນ
            <option value="#{vs.index}" ${vs.index==selectedSemester ? 'selected':''}>#${semester}</option>
        </c:if>           ↗ ຖື index
    </c:forEach>
</select>

```

ໃຫ້ servlet ທີ່ດັ່ງນັບດົງໄປ !

```

@WebServlet(
    name = "CourseListServlet",
    value = {"course-list"}
)
public class CourseListServlet extends HttpServlet {
    public CourseListServlet() {
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        request.setAttribute("semesters", Semester.getAllSemesterText());
        this.getServletContext().getRequestDispatcher("/CourseList.jsp").forward(request, response); //ໃຫ້ course list
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        Map<String, String[]> parameterMap = request.getParameterMap(); ເຫັນມູນຫ່າຍມາສໍາຄັນ
        request.setCharacterEncoding("UTF-8"); * ຢັດກາໄຟ່ນກາບໄກນ
        if (parameterMap.get("semester") == null) { ກໍາໄລສໍ່ແປງວ່າໃນກໍາສັ່ງ semester ມີເບີນກລົມຈັງ course list (doGet())
            this.doGet(request, response);
        } else { , ມີມີນດຳ array
            int semester = Integer.valueOf((String[])parameterMap.get("semester"))[0];
            request.setAttribute("semesters", Semester.getAllSemesterText());
            request.setAttribute("selectedSemester", semester); ດູວອນກ່າວນທີ່user ເລືອດເລັກນັ້ນໄຫວ
            request.setAttribute("subjects", CourseRepository.getSubjects(semester)); ວຽກມີ 1 ຕົວ collection
            this.getServletContext().getRequestDispatcher("/CourseList.jsp").forward(request, response);
        }
    }
}

```

ໃຫ້ model (JAVA)

```

public class Semester {
    private static final String[] TITLE = new String[]{null, "ການ 1/ ປຶກສຶກຂາທີ 1", "ການ 2/ ປຶກສຶກຂາທີ 1",

    public Semester() {
    }

    public static String[] getAllSemesterText() { return TITLE; }

    public static String getSemesterText(int semesterNumber) {
        return semesterNumber >= TITLE.length ? null : TITLE[semesterNumber];
    }
}

```

```

<c:if test="${subjects != null}">
    <div class="container m-auto h-auto">
        <form action="register" method="post">
            <input type="hidden" name="semester" value="${selectedSemester}"> ↑ ชี้ว่าที่ต้องการส่งกลับไป
            <div class="row bg-white">
                <div class="col-1">ลำดับ</div>
                <div class="col-1">รหัส</div>
                <div class="col-5">ชื่อวิชา</div>
                <div class="col-1">หน่วยกิต</div>
                <div class="col-1">เลือก</div>
            </div>
            <c:forEach items="${subjects}" var="subject" varStatus="vs">
                <div class="row bg-transparent">
                    <div class="col-1">${vs.count}</div>
                    <div class="col-1">${subject.subjectId}</div>
                    <div class="col-6">${subject.title}</div> ↑ ที่พิมพ์มาแล้วกันดี
                    <div class="col-1">${subject.credit}</div> ↑ registeredSubject
                    <div class="col-1">
                        <input type="checkbox" name="registeredSubject" value="${subject.subjectId}"> ↑ เมนูแบบ checkbox ก็จะได้ตัว multiple value
                    </div>
                </div>
            </c:forEach>
            <hr>
            <div class="form-group"><input type="submit"></div>
        </form>
    </div>
</c:if>

```

JSP

```

@WebServlet(
    name = "RegisterCourseServlet",
    value = {"register"}
)
public class RegisterCourseServlet extends HttpServlet {
    public RegisterCourseServlet() {
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        Map<String, String[]> parameterMap = request.getParameterMap(); // แปลง成 map value เป็น array
        int semester = Integer.valueOf(((String[])parameterMap.get("semester"))[0]); // ① ตัวแปรที่ร่วมกันระหว่างหน้าเบื้องหน้า
        * HttpSession session = request.getSession(); // เก็บค่าไว้ใน session
        CourseRegistered courseRegistered = (CourseRegistered)session.getAttribute(s: "courseRegistered");
        if (courseRegistered == null) { // เช็คต่อเมื่อแรกๆ
            courseRegistered = new CourseRegistered(); // ส่วนของข้อมูลจะมาใส่
            session.setAttribute(s: "courseRegistered", courseRegistered); ↑ ผูกของตัวอยู่บ้าง?
        } else {
            courseRegistered.removeAllRegisteredCourse(semester); getSession → vo session
            getAttribute → vo voใน Session
        }

        String[] var7 = (String[])parameterMap.get("registeredSubject"); // ② ตัวแปรที่ส่งมาแบบ multiple value
        int var8 = var7.length;

        for(int var9 = 0; var9 < var8; ++var9) {
            String subjectId = var7[var9]; ↑ หัวข้อเพื่อตรวจสอบว่าต้องการส่งกลับ!
            courseRegistered.registerSubject(semester, CourseRepository.getSubject(semester, subjectId));
            System.out.println(CourseRepository.getSubject(semester, subjectId));
        }

        System.out.println(); ↑ จบการเขียนผลลัพธ์
        this.getServletContext().getRequestDispatcher(s: "/index.jsp").forward(request, response);
    }
}

```

JPA

(Java Persistence API)

- layer ที่ชื่อ JPA ที่มี interface สำหรับการจัดการข้อมูล
- ฟังก์ชัน library , dependency
- ต้องมีตัวกลางฐานข้อมูล

(Auxiliary Device / External memory)

* Secondary Storage

- เอกชนุลักษณ์
- เป็นได้ทั้ง input + output
- กิ่งก้าน
- database
- Persistence data (เก็บกาว)

└ 临时数据 transient data

COMPUTER SYSTEMS

- เป็นไปตามแนวคิดของ Von Neumann

- ห้าใจสำคัญ! 1) data 2) instructions 2 คำสั่งทั้งหมดอยู่ใน main memory

└ เช่น 1 statement อาจแทนไปได้หลาย instructions
method เป็น instructions

* obj เป็นทั้ง data + instruction



class ที่ใช้เก็บข้อมูล
จาก table เราเรียก
กันว่า "Entity"

ORM (Object Relational Mapping) 乃即把 row - object

- 1 class สร้างเก็บลง obj เท็อน database (ข้อมูล 1 row = 1 object)

db (Entity Relation ผังงาน)

- primary key
- foreign key

(one to many)

Person 1 : m Address เช่น code ปะจัง?

class Person { Address address[]; }

or class Person { List<Address> addresses; }

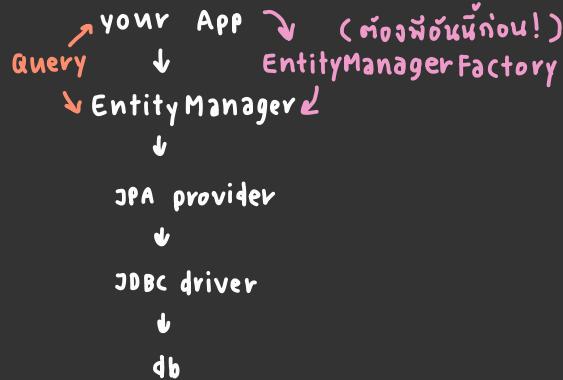
python (Object Relationship)

- Aggregation
- Inheritance

JPA 乃 package javax.persistence

- EntityManager Factory
 - Entity Transaction
 - EntityManager
 - Query
 - Persistence

JPA CLASS ARCHITECTURE



enterprise vendor (ที่ library ไม่ใช่) - Hybernate, Eclipselink, Toplink, Spring Data JPA

ENTITIES (ព័ត៌មានកុំគោលចាយពីការបង្កើត) *

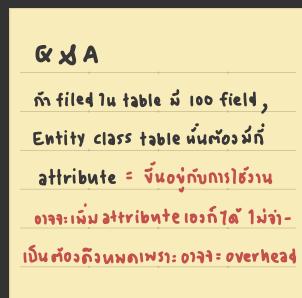
- plain old java object (POJO) java class ដែលមានរាយការណ៍ (Getter, Setter) នៅក្នុងការបង្កើត Entity ទាំងអស់
- class represent a table in a relational db
- Instances(java) = row(db) [class person ម៉ោង table person] + នៅក្នុង annotation នៃលោខាងក្រោមនេះ គឺជា entity
- Requirement
 - import ពី javax.persistence.Entity
 - class ត្រូវបាន public or protected (X private), ត្រូវបាន default constructure
 - អាមេរិយៈ final *

```
@Entity
@Table(name="USER")
public class User {
    @Id
    @GeneratedValue(strategy=GenerationType.SEQUENCE, generator="UserSequence")
    private Long id; // primary key

    @Column(name="USERNAME", nullable=false)
    private String username;

    @Column(name="FULL_NAME")
    private String fullName;
    ...
}
```

| USER TABLE | | |
|------------|----------|----------------|
| ID | USERNAME | FULL_NAME |
| 1 | eatrocks | Bruce Campbell |
| 2 | mark | Mark Jones |



EX A

ការ map ពី table ដែលមាន 100 field, Entity class table ដែលមាន 3 field
attribute = ចំណាំការបង្កើត
បញ្ជាផែន attribute ទេរកទៀត នៅក្នុង
បែងចែកតួនាទីការងារ: overhead

ENTITY MANAGER

- create, remove និងកែតាំង
- find ឬស្វែនតារាបត្រ
- សរែច queries ។

Ex User user = entityManager.find(User.class, 1002);

create

L មិនមែន method នៃ class persistence javax.persistence.Persistence

L root class voi EntityManager

L បានចូលរួមនៅក្នុង persistence unit (persistence.xml)

L ត្រូវបានចូលរួមនៅក្នុង driver db

1) ចូលរួមនៅក្នុង persistence unit

2) ជាក៏ class persistence ។ ប្រើប្រាស់ method ដោយស្វែន EntityManagerFactory

3) សរែច EntityManager

- EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence unit name from persistence.xml");
- EntityManager em = emf.createEntityManager();

```
User user = em.find(User.class, 1002);
Customer c = em.find(Customer.class, 2001);
Subject subject = em.find(Subject.class, "INT202");
Product product = em.find(Product.class, "S12_101");
Employee emp = em.find(Employee.class, 1002);
```

ចូលរួមនៅក្នុង path

```
<persistence-unit name="default">
    <class>sit.int202.classicmodelweb.entities.Office</class>
    <properties>
        <property name="jakarta.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/> 1) driver
        <property name="jakarta.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/classicmodels"/> 2) url
        <property name="jakarta.persistence.jdbc.user" value="root"/> 3) username
        <property name="jakarta.persistence.jdbc.password" value="143900"/> 4) password
    </properties>
</persistence-unit>
```

persistence.xml