

메타웍스 프레임워크

Open Cloud Engine Initiative

Rick Jinyoung Jang

www.opence.org



OPEN CLOUD ENGINE

What is Metaworks?

1. Metadata Oriented Application Framework
2. Developed by Jinyoung Jang in 1998
3. Inspired from the Adaptive Object Models and OMG Reflection, MDA
4. Application Component Generation on the fly from metadata
5. Metaworks1: Generates database query and action, Plain HTML written in ASP - 1998
6. Metaworks2: Generates Swing User interfaces written in Java, is adopted in uEngine Process Designer – 2003
7. Metaworks3: Generates Client-side rendering supported Web Uis. Metadata definition by annotation – 2011

So Metaworks3...

1. Is A POJO framework
2. Encourages the Domain-Driven Design
3. Especially for developing model-driven applications (UML, BPMN, etc)

What is Framework?

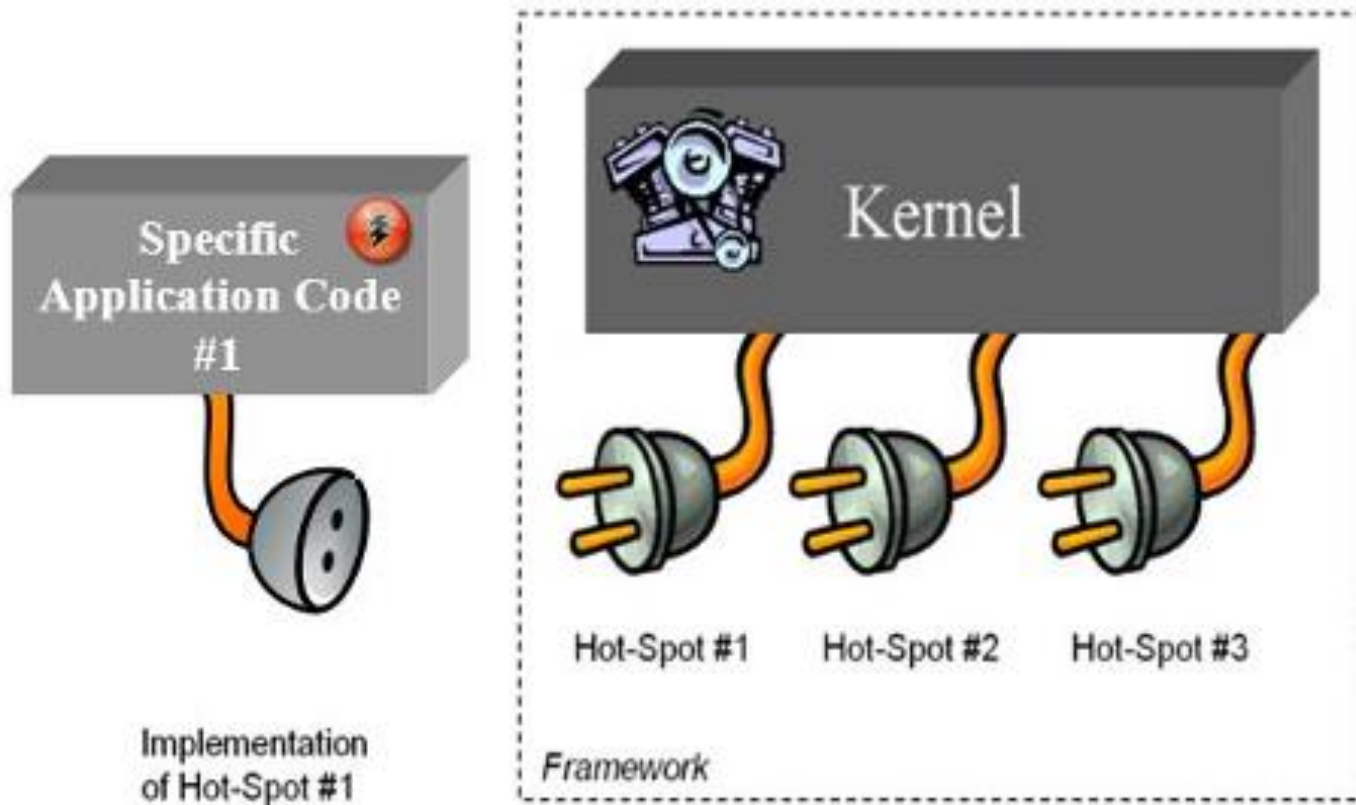
The background of the slide is a photograph of a construction site. It shows a complex network of metal scaffolding against a clear blue sky. Several workers in safety gear are visible, climbing and working on the structure. The scaffolding is made of vertical and horizontal poles connected by diagonal bracing, creating a grid-like pattern.

Achieving 'Reuse of Design'

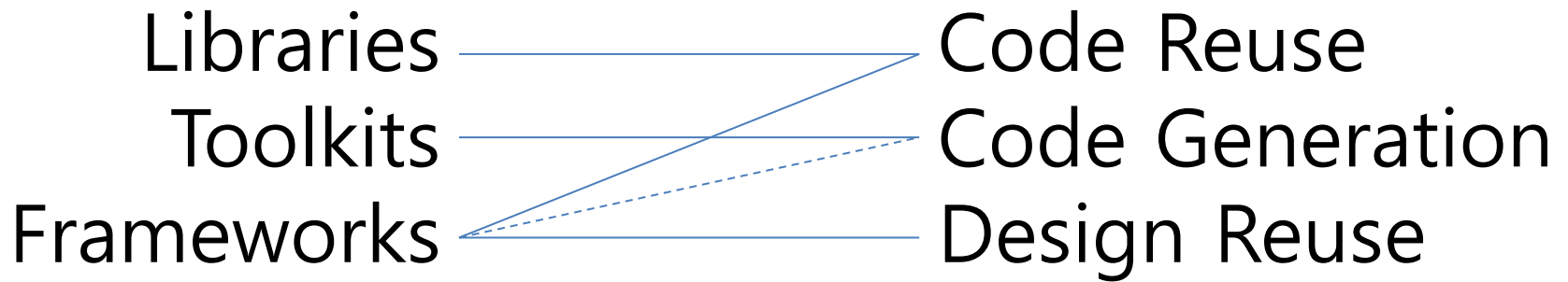
through

- 1. Inversion of Control**
- 2. Separation of Concerns**

Framework of Framework

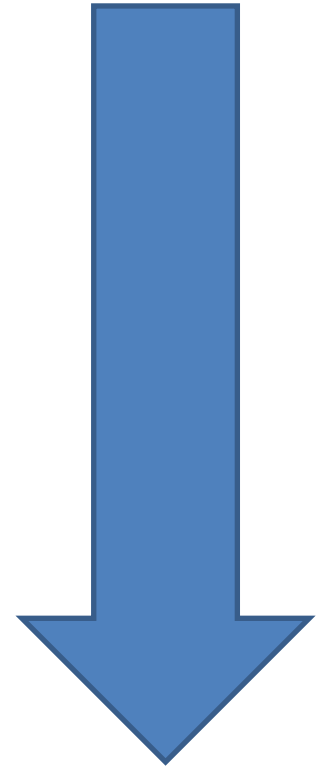


Frameworks, Toolkits and Libraries



History of OO Frameworks and Methodologies

1. Smalltalk
2. STL (Standard Template Language) of C++
3. Adaptive Object Models – Ralf Johnson
4. Design Patterns – Gang of four
5. Object-Oriented Frameworks
6. OMG Reflection
7. Java – 1995
8. IBM's Sanfrancisco Framework
9. Java 2 Enterprise Edition – EJB 2.1
10. EJB Patterns



- *Bigger and Complex Super Classes*
- *Java doesn't Support Multiple Inheritance*

Domain Specific Application Frameworks

Example of Business Domain Specific Application Frameworks :

1. Car Simulation
2. Product Simulation
3. Or your Problem itself!!
4. uEngine is also a DSAF!
→ abstracts business problems

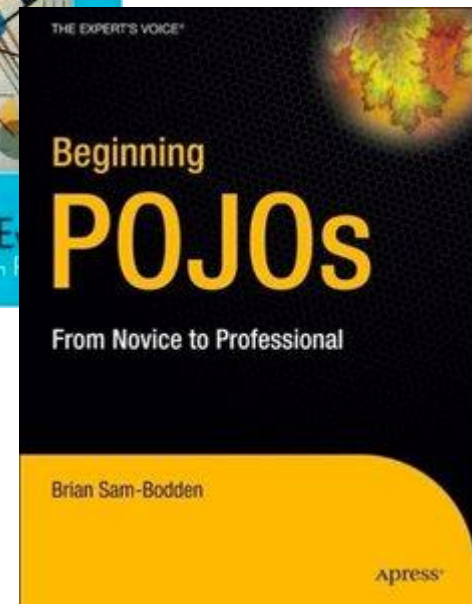
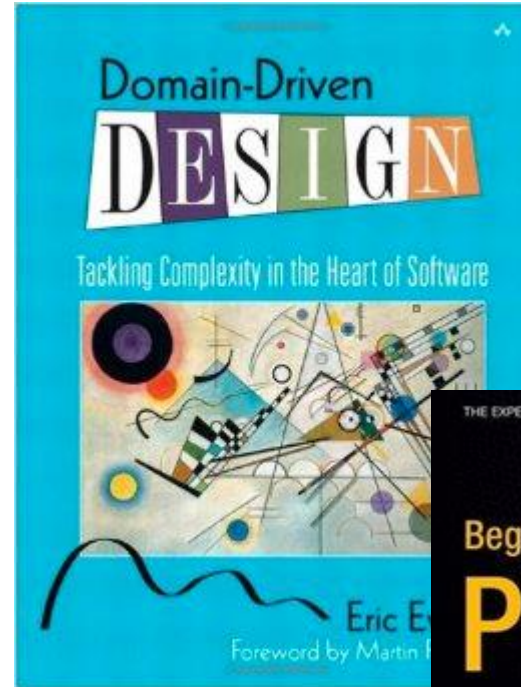
Example of System Domain Specific Application Frameworks :

1. EJB
2. Struts
3. Swing
4. Apache Mesos
→ abstracts technical problems

But We Can't Inherit Two Super Classes at the same time!!!

Return to the “Pure Java”

- Aspect Oriented Programming
- Java Annotation
- JPA
- JAX-RS
- EJB 3.1



EJB 2.1 vs EJB 3.1

Java 클래스

```
public class AccountBean
implements javax.ejb.SessionBean {

    SessionContext ctx;
    DataSource accountDB;

    public void setSessionContext(SessionContext ctx) {
        this.ctx = ctx;
    }

    public void ejbCreate() {
        accountDB = (DataSource)ctx.lookup(
            "jdbc/accountDB");
    }

    public void ejbActivate() { }
    public void ejbPassivate() { }
    public void ejbRemove() { }

    public void setAccountDeposit(int empId,
                                   double deposit) {
        ...
        Connection conn = accountDB.getConnection();
        ...
    }
    ...
}
```

Java 클래스

```
@Stateless
public class AccountBean implements Account
{
    @Resource private DataSource accountDB;

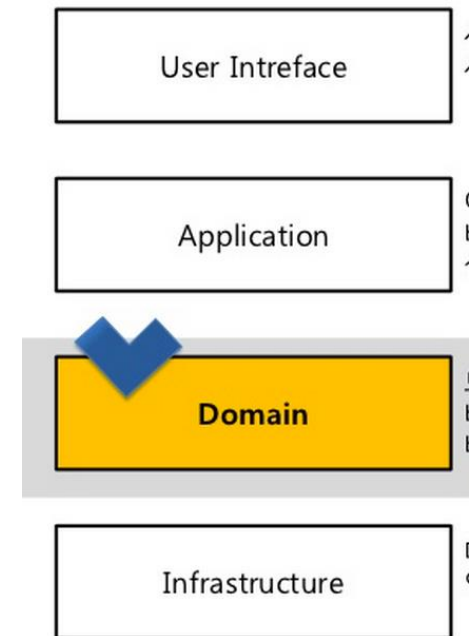
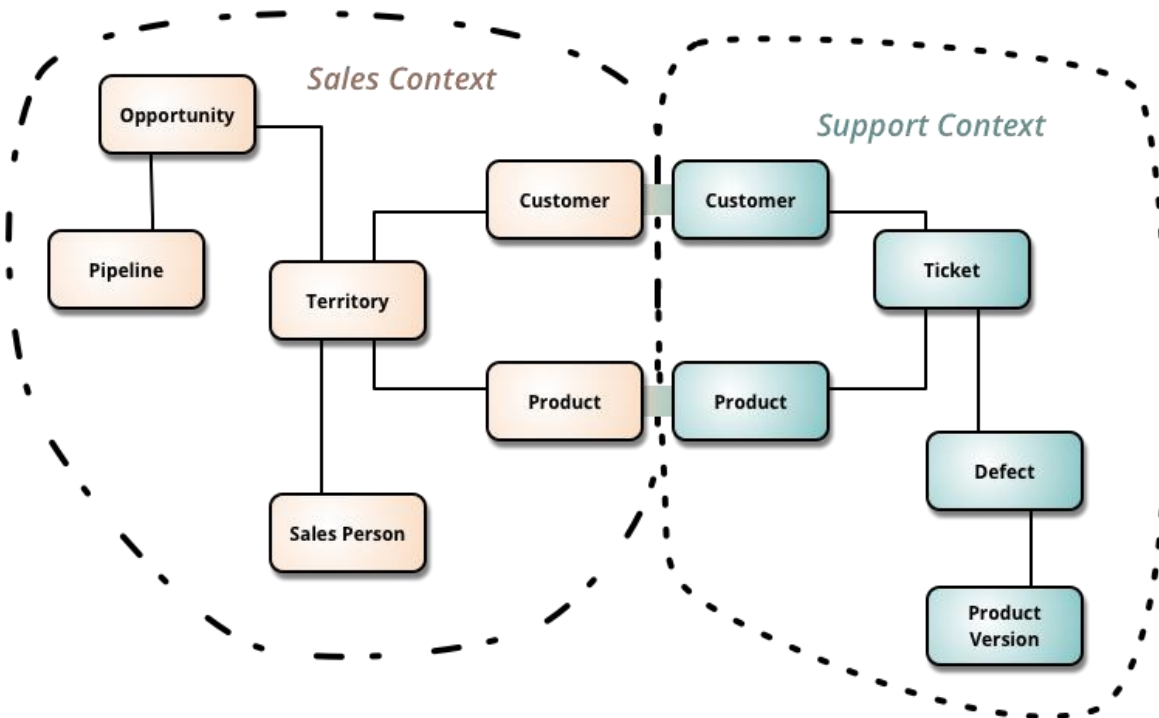
    public void setAccountDeposit(int customerId,
                                   double deposit) {
        ...
        Connection conn = accountDB.getConnection();
        ...
    }
    ...
}
```

출처: [http://www-](http://www-01.ibm.com/support/knowledgecenter/SS8PJ7_9.1.0/com.ibm.javaee.doc/topics/cejb3vejb21.html?cp=SS8PJ7_9.1.0%2F1-1-0-1)

[01.ibm.com/support/knowledgecenter/SS8PJ7_9.1.0/com.ibm.javaee.doc/topics/cejb3vejb21.html?cp=SS8PJ7_9.1.0%2F1-1-0-1](http://www-01.ibm.com/support/knowledgecenter/SS8PJ7_9.1.0/com.ibm.javaee.doc/topics/cejb3vejb21.html?cp=SS8PJ7_9.1.0%2F1-1-0-1)

Domain-Driven Design

1. Using 'Ubiquitous Language' so that the domain expert and developer can communicate
2. Bounded-Context is principal for dividing packages. ... and there are many practices to keep the domain model out of the layer



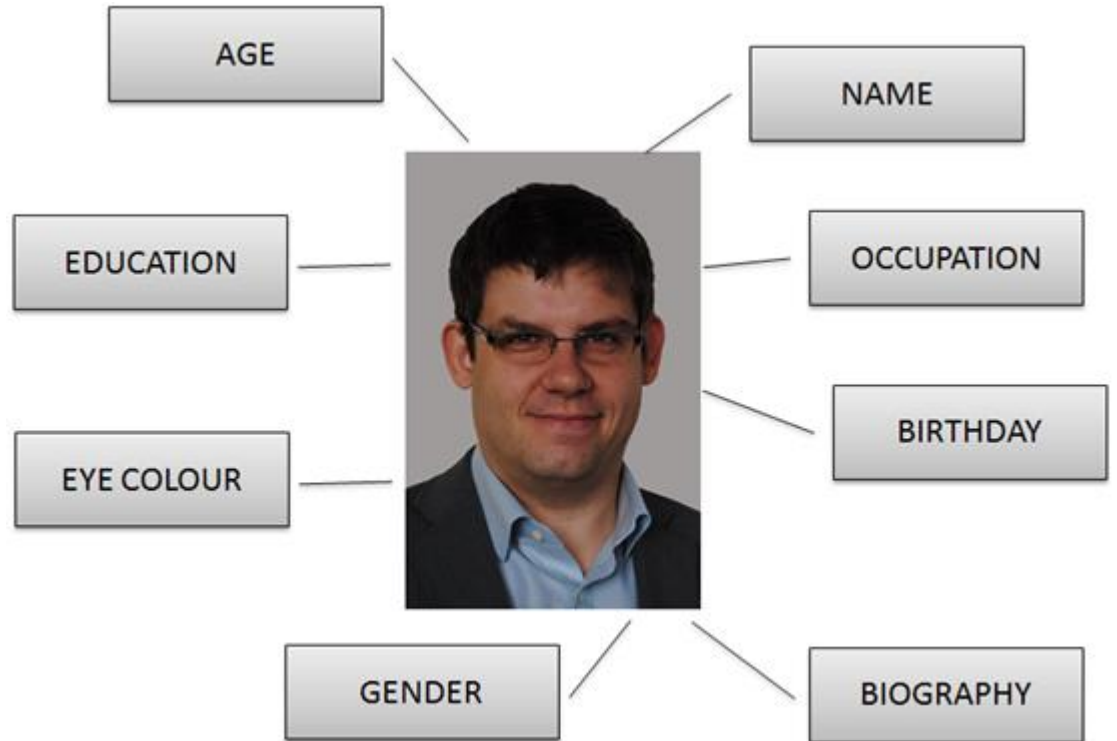
LAB Time:

Design of BSS Framework

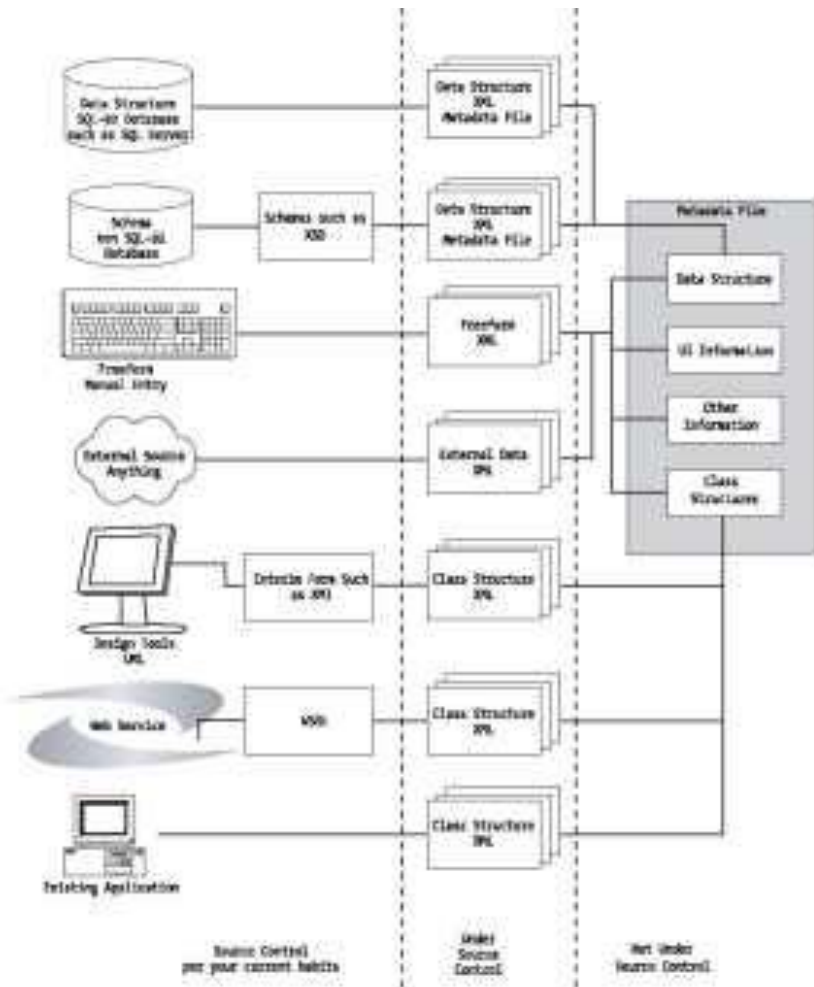
What is Metadata?

"Knowing yourself is the beginning of all wisdom."

~Aristotle



Scattered Metadata in Layered Architecture



- Schema definitions such as XSD
- Databases such as SQL Server
- Manually entered freeform metadata
- Web Services via Web Service Description Languages (WSDLs)
- External sources, such as mainframes, Excel, and so on
- Design tools such as Unified Modeling Language (UML)
- Existing applications and source code

Figure 2-2 Metadata sources

<http://www.aspfree.com/c/a/database/extracting-metadata/>

An infinite loop



ANATOMY
TRAINS

ISOLATION

VS

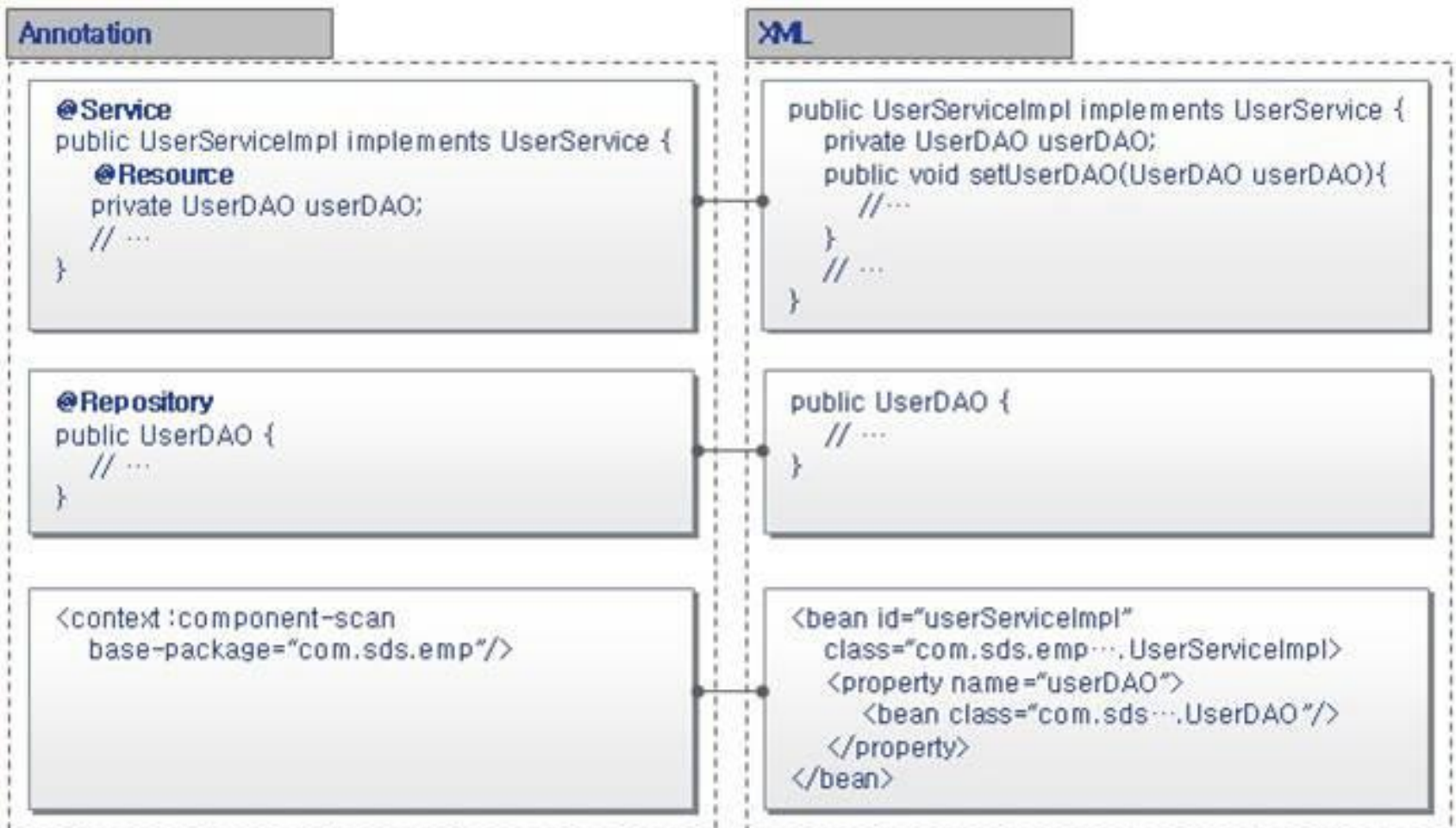
INTEGRATION



VS



e.g. Spring's Configuration



Put the metadata in one place.
Where? Into Domain Class

LAB Time:

JPA metadata injection to the BSS
Framework

What is Metaworks?

1. Metadata Oriented Application Framework
2. Developed by Jinyoung Jang in 1998
3. Inspired from the Adaptive Object Models and OMG Reflection, MDA
4. Application Component Generation on the fly from metadata
5. Metaworks1: Generates database query and action, Plain HTML written in ASP - 1998
6. Metaworks2: Generates Swing User interfaces written in Java, is adopted in uEngine Process Designer – 2003
7. Metaworks3: Generates Client-side rendering supported Web Uis. Metadata definition by annotation – 2011

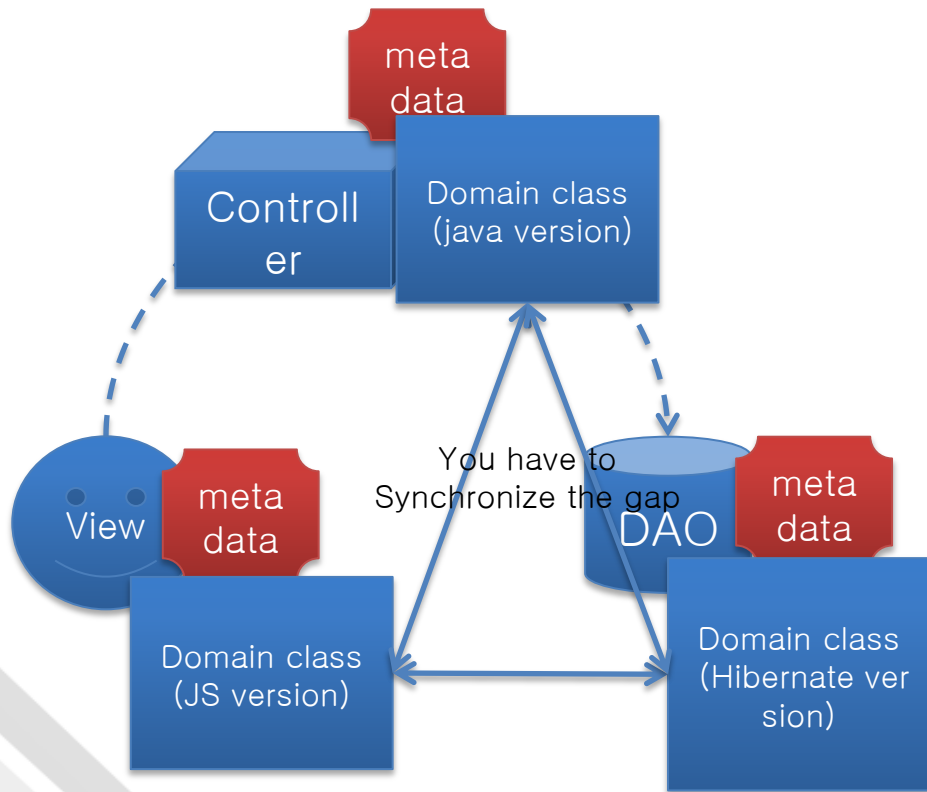
So Metaworks3...

1. Is A POJO framework
2. Encourages the Domain-Driven Design
3. Especially for developing model-driven applications (UML, BPMN, etc)

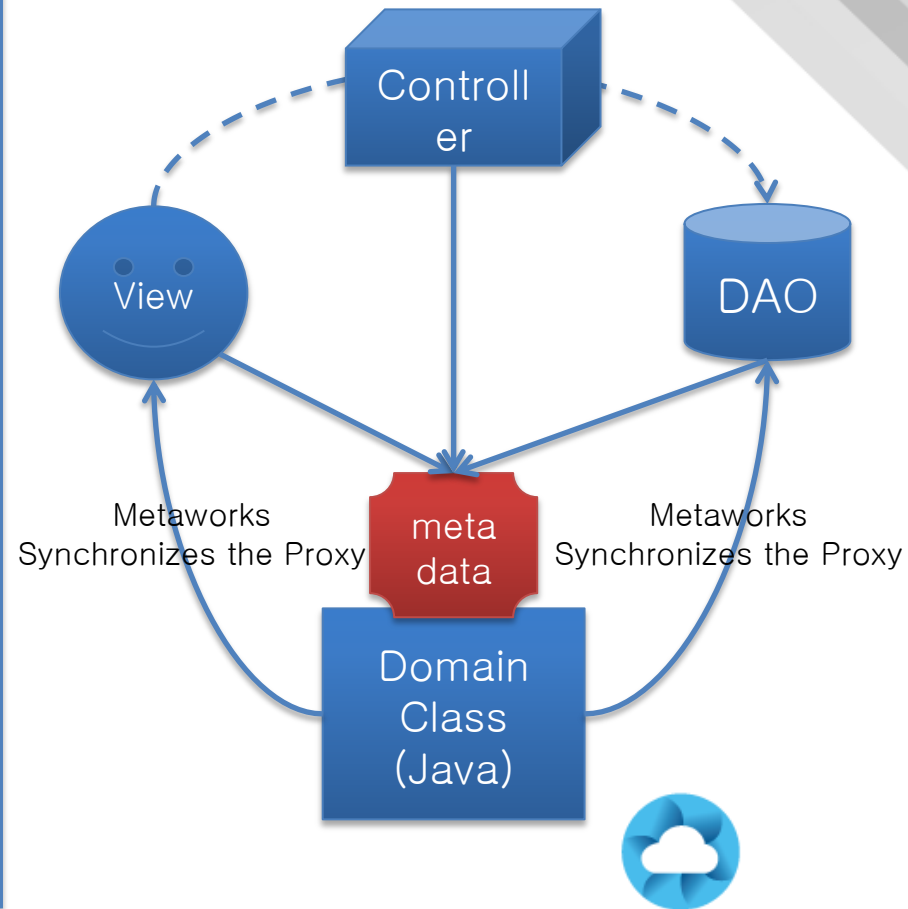
메타웍스 동작 메커니즘:

- 흠어진 메타데이터를 자동으로 관리하는 메타웍스3

General Approach:
Spring MVC, JSON, jQuery, Hibernate



Using Metaworks3:



"메타웍스3 우산"



< 선언형 프로그래밍 모델과 추상성 극대화를 통한 생산성 >

- 프로그래밍 모델 추상화
 - 자바-자바스크립트 간 통신 추상화
 - 자바-자바스크립트 간 메타데이터 자동 동기화
 - 자바스크립트 인터랙션 (버튼, 키보드/마우스) 추상화
 - 네비게이션 추상화
 - 룡-폴링 기반 콜백 추상화
- 데이터베이스 접근 추상화
 - JPA어노테이션 기반 ORMMapping
 - 데이터베이스 캐시 / 동기화
 - 트랜잭션 자동화 (스프링없이 동작가능)

< 테스트 자동화와 품질 향상 >

- 매우 짧은 코드 – 품질에 직결
- 설계가 곧 구현 – 모델중심의 생산성과 품질
- 아규먼트 없는 서비스메서드 – 객체의 응집도를 유지
- 웹 기반 테스트 자동화
- Guided Tour 기능 자동화



Metaworks Basic Annotations

- Field descriptors:
 - @Id
 - @Range
 - @Face
- Method descriptors:
 - @ServiceMethod



LAB Time: Adding metaworks annotation to BSS Framework



OPEN CLOUD ENGINE

메타웍스3 프로젝트 설치 및 실행

1. 사전 준비 사항

1. Maven 3.0 이상
2. Eclipse or IntelliJ (추천)
3. <http://wiki.opencloudengine.org/display/MET/Metaworks3+Home> 사이트 접속

2. 메타웍스3 기본 프로젝트 구조 생성

1. Maven Archetype을 이용하여 프로젝트 기본 구조를 생성:

```
mvn archetype:generate -DarchetypeGroupId=org.uengine.metaworks -  
DarchetypeArtifactId=metaworks-sample-archetype -DarchetypeVersion=1.0.0 -  
DarchetypeRepository=http://maven.opencloudengine.org/content/repositories/releases
```

2. [Eclipse 에서] Import As Maven Project, [IntelliJ에서] 그냥 Open...

3. 설정 파일 구성

[applicationContext.xml, dwr.xml]

3. 메타웍스3 프로젝트 실행

1. WAS 기동

2. 웹 브라우저 : <http://localhost:8080/index.html>

메타웍스3 프로젝트 설치 및 실행

3. 클래스를 생성하고 바로 테스트를 할 수 있는 화면

<http://localhost:8080/runner.html?className=풀패키지명>

e.g. <http://localhost:8080/runner.html?className=Login>

Advanced Annotations

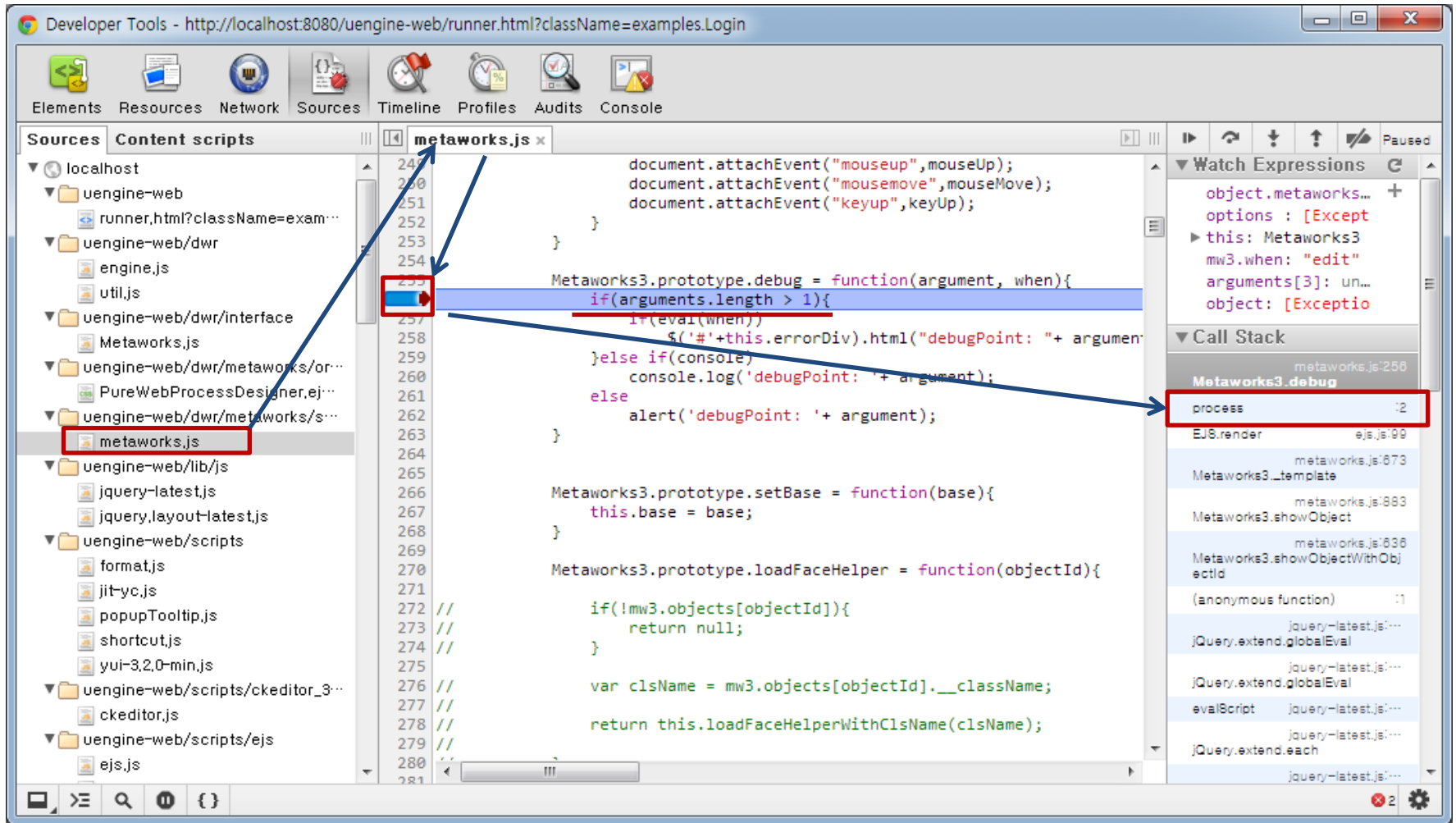
실전7. 디버깅하기 - ejs

1단계 - 디버깅할 위치에 디버그 코드를 작성한다.

```
<%  
    mw3.debug(true);  
%>  
  
<table border="0" cellspacing="0" >  
  <tbody>  
    <tr>  
      <th scope="row">ID</th>  
      <td> <%=fields.userId.here()%>  
    </tr>  
    <tr>  
      <th scope="row">비밀번호</th>  
      <td> <%=fields.password.here()%>  
    </tr>  
  </tbody>  
</table>  
  
<table border="0" cellspacing="0" >  
  <tbody>  
    <tr>  
      <td> <%=methods.login.here()%> </td>  
    </tr>  
  </tbody>  
</table>
```

실전7. 디버깅하기 - ejs.js

2단계 - 디버그 코드에 Breakpoint 를 설정



실전7. 디버깅하기 - ejs.js

3단계 - Call Stack 의 process 를 클릭하여 디버깅

Developer Tools - http://localhost:8080/uengine-web/runner.html?className=examples.Login

Elements Resources Network Sources Timeline Profiles Audits Console

Sources Content scripts

localhost

- uengine-web
 - runner.html?className=exam...
 - uengine-web/dwr
 - engine.js
 - util.js
 - uengine-web/dwr/interface
 - Metaworks.js
 - uengine-web/dwr/metaworks/or...
 - PureWebProcessDesigner.ej...
 - uengine-web/dwr/metaworks/s...
 - metaworks.js
 - uengine-web/lib/js
 - jquery-latest.js
 - jquery.layout-latest.js
 - uengine-web/scripts
 - format.js
 - jit-yc.js
 - popupTooltip.js
 - shortcut.js
 - yui-3.2.0-min.js
 - uengine-web/scripts/ckeditor_3...
 - ckeditor.js
 - uengine-web/scripts/ejs
 - ejs.js

metaworks.js (program) x

```
1 /*uengine-web/dwr/metaworks/examples/ILogin.ejs?ver=2*/this.process = function() {
2   mw3.debug(true);
3   mw3.template_line = 3; __View0.push("  \n");
4   mw3.template_line = 4; __View0.push("  \n");
5   mw3.template_line = 5; __View0.push("<table border=\"0\" cellspacing=\"0\" >");
6   mw3.template_line = 6; __View0.push("  <tbody>\n");
7   mw3.template_line = 7; __View0.push("    <tr>\n");
8   mw3.template_line = 8; __View0.push("      <th scope=\"row\">ID</th>\n");
9   mw3.template_line = 9; __View0.push("    <td>); mw3.template_line = 9;
10  mw3.template_line = 10; __View0.push("  </tr>\n");
11  mw3.template_line = 11; __View0.push("  <tr>\n");
12  mw3.template_line = 12; __View0.push("    <th scope=\"row\">비밀번호</th>");
13  mw3.template_line = 13; __View0.push("    <td>); mw3.template_line = 13;
14  mw3.template_line = 14; __View0.push("  </tr>\n");
15  mw3.template_line = 15; __View0.push("  </tbody> \n");
16  mw3.template_line = 16; __View0.push("</table>\n");
17  mw3.template_line = 17; __View0.push("\n");
18  mw3.template_line = 18; __View0.push("<table border=\"0\" cellspacing=\"0\" >");
19  mw3.template_line = 19; __View0.push("  <tbody>\n");
20  mw3.template_line = 20; __View0.push("    <tr>\n");
21  mw3.template_line = 21; __View0.push("      <td>); mw3.template_line = 21;
22  mw3.template_line = 22; __View0.push("    </tr>\n");
23  mw3.template_line = 23; __View0.push("  </tbody>\n");
24  mw3.template_line = 24; __View0.push("</table>\n");
25  ;return __View0.join('');}}catch(e){throw e}};
```

Watch Expressions

- object.metaworks...
- options : Object
- this : Object
- mw3.when: "edit"
- arguments[3]: un...
- object: [Exceptio

Call Stack

- metaworks.js:256 Metaworks3.debug
- process :2**
- EJS.render ejs.js:99
- metaworks.js:673 Metaworks3._template
- metaworks.js:883 Metaworks3.showObject
- metaworks.js:636 Metaworks3.showObjectWithObj
- ectId
- (anonymous function) :1
- jquery-latest.js:... JQuery.extend.globalEval
- jquery-latest.js:... JQuery.extend.globalEval
- jquery-latest.js:... evalScript
- jquery-latest.js:... JQuery.extend.each
- jquery-latest.js:...

Metaworks Development Process

1. (Optional) UML을 이용한 설계
2. 도메인 자바 코드의 생성
3. 타 POJO Framework의 적용 - JPA를 통한 ORM 생성, JAX-RS를 통한 서비스 공개
4. Metaworks3를 통한 Default Application 생성
5. UI 구체화 / 커스터마이징
6. Round-trip

UI interaction control by metaworks

법칙 1: 객체 매핑

로그인 모듈에 대한 객체 매핑 방법

The screenshot shows the Facebook login page with several annotations for object mapping in a Java application:

- 이메일 (Email):** A green box highlights the email input field. A red dashed arrow points from this box to the `String email` property in the `Login.java` class.
- 비밀번호 (Password):** A yellow box highlights the password input field. A red dashed arrow points from this box to the `String password` property in the `Login.java` class.
- 로그인 (Login):** A red box highlights the login button. A red dashed arrow points from this box to the `Main login()` method in the `Login.java` class.
- 속성은 프로퍼티로 (Properties are Properties):** A red box highlights the text "속성은 프로퍼티로" (Properties are Properties) on the left side of the page. A red dashed arrow points from this box to the `String email` property in the `Login.java` class.
- 행위는 메서드로 (Actions are Methods):** A red box highlights the text "행위는 메서드로" (Actions are Methods) on the right side of the page. A red dashed arrow points from this box to the `Main login()` method in the `Login.java` class.

The `Login.java` class is shown in a blue box on the left:

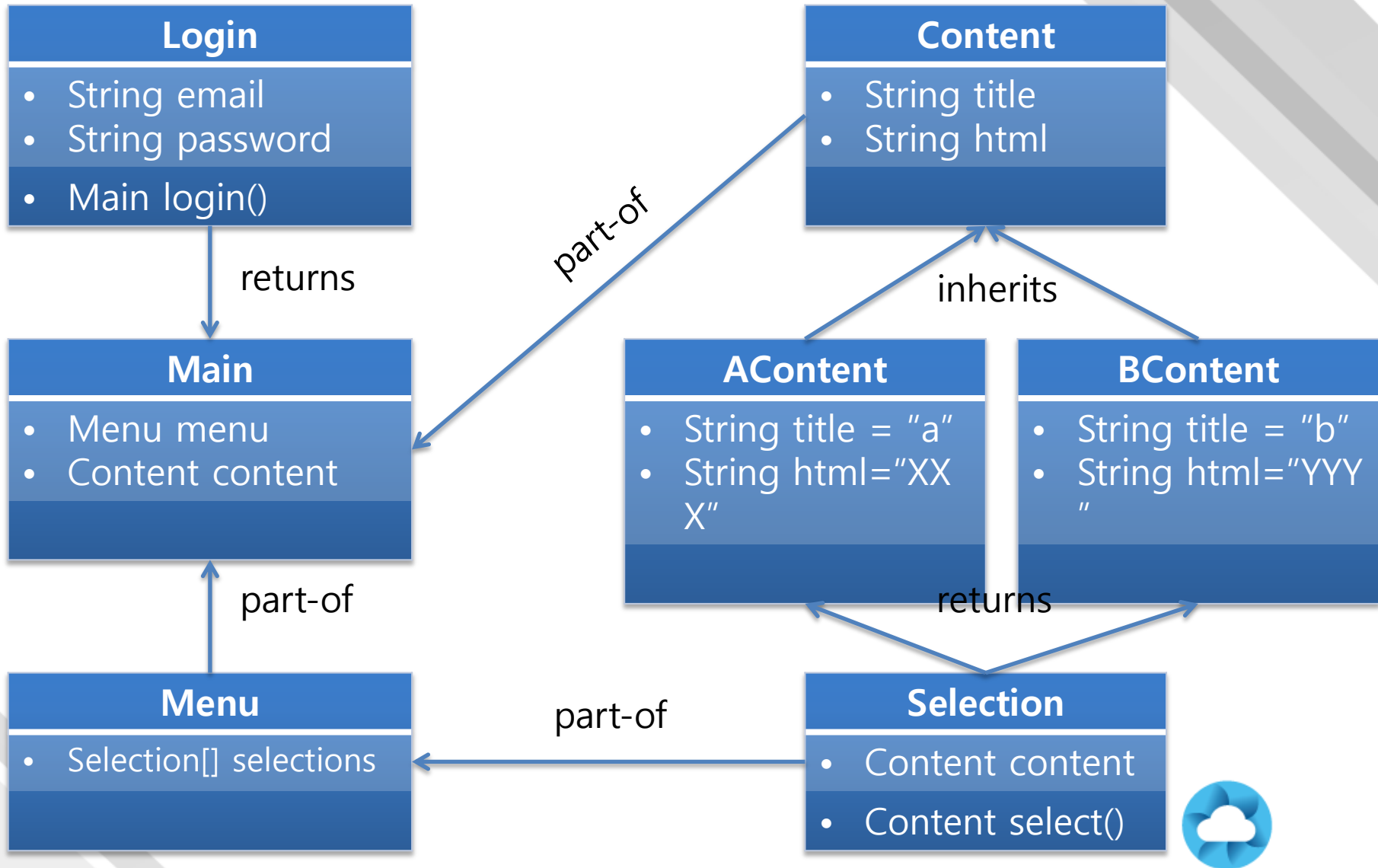
```
Login.java
```

- String email
- String password
- Main login()

The Facebook page also shows the "가입하기" (Sign Up) section with a form for creating a new account. The form includes fields for name, email, and password, and a "가입하기" button. A red dashed arrow points from the "가입하기" button to the "로그인" button.

At the bottom of the page, there is a footer with language options: 한국어, English (US), Español, Português (Brasil), Français (France), Deutsch, Italiano, العربية, हिन्दी, 中文(简体), and a link to "페이지 만들기" (Create Page).

메타웍스3 프로그래밍 모델 - 전체



Login.java

```
1 package org.metaworks.example.navigation;
2
3 import org.metaworks.annotation.ServiceMethod;
4
5 public class Login {
6
7     String userId;
8     public String getUserId() {
9         return userId;
10    }
11    public void setUserId(String userId) {
12        this.userId = userId;
13    }
14
15    String password;
16    public String getPassword() {
17        return password;
18    }
19    public void setPassword(String password) {
20        this.password = password;
21    }
22
23    @ServiceMethod(callByContent=true)
24    public Main login() throws Exception{
25        return new Main();
26    }
27 }
28
```

Setter/getter가
있으면 '프로퍼티'
라고 하며, 웹상
에서는 입출력이
요망되는 주요데
이터가 된다

행위는 일반적
내부 행위와 웹
에서 출력될(버튼)
행위와의 구분을
위하여
@ServiceMethod
애노테이션을
준다

법칙2: 리턴 행위는 객체의 변화를 암시한다.

Login.java

UserId

Password

return
new Main()
;

login() 이 실행되면
Main을 리턴하므로
Main을 화면에 그려라!

Main.java



OPEN CLOUD ENGINE

Main.java



Menu object

MenuItem object

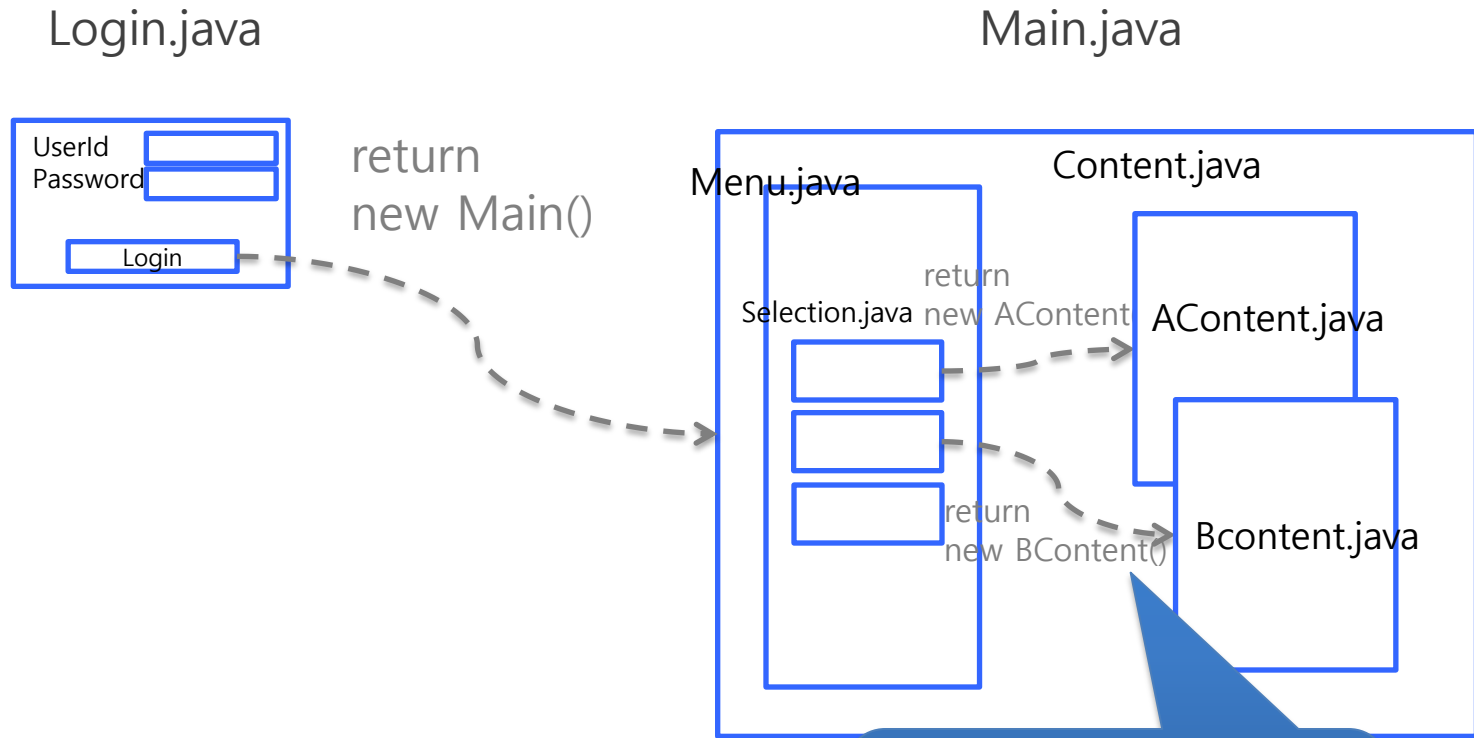
Contents object

Main.java

```
1 package org.metaworks.example.navigation;
2
3 public class Main {
4
5     Menu menu;
6     public Menu getMenu() {
7         return menu;
8     }
9     public void setMenu(Menu menu) {
10         this.menu = menu;
11     }
12
13     Content content;
14     public Content getContent() {
15         return content;
16     }
17     public void setContent(Content content) {
18         this.content = content;
19     }
20
21     protected Main(){
22         setMenu(new Menu());
23         setContent(new Content());
24     }
25
26 }
27
```

protected 로 생
성자가 보안처리
되었기 때문에
앞서
Login.login()을
통하지 않고서는
진입할 수 없게
된다.

법칙3 : 리턴객체는 가장 편한 곳으로 자리한다.



(화면에 여러 객체가 이미 존재하는 경우) 리턴된 객체는 자신이 가장 부합되는 응집력을 가진 화면 요소에 가서 그려짐



Menu.java

```
1 package org.metaworks.example.navigation;
2
3 public class Menu {
4
5     protected Menu(){
6         Selection selection1 = new Selection();
7         selection1.setContent("AContent");
8
9         Selection selection2 = new Selection();
10        selection2.setContent("BContent");
11
12        setSelections(new Selection[] {selection1, selection2});
13    }
14
15    Selection[] selections;
16    public Selection[] getSelections() {
17        return selections;
18    }
19    public void setSelections(Selection[] selections) {
20        this.selections = selections;
21    }
22
23 }
24
```

Selection.java

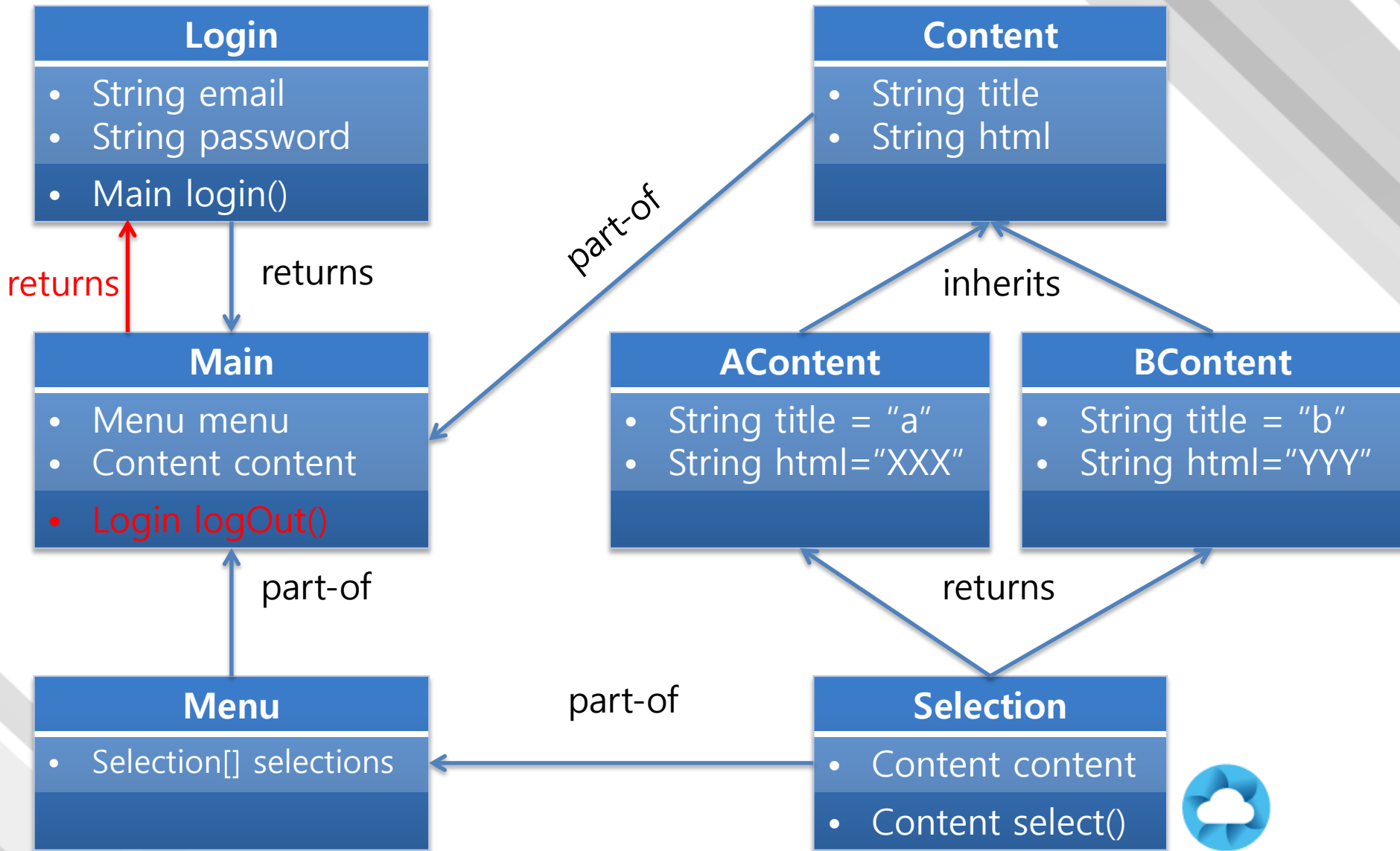
```
1 package org.metaworks.example.navigation;
2
3 import org.metaworks.annotation.Id;
4 import org.metaworks.annotation.ServiceMethod;
5
6 public class Selection {
7     String content;
8     @Id
9     public String getContent() {
10         return content;
11     }
12     public void setContent(String content) {
13         this.content = content;
14     }
15
16     @ServiceMethod
17     public Content select() throws InstantiationException, IllegalAccessException, ClassNo
18         if("AContent".equals(content))
19             return new AContent();
20         else if("BContent".equals(content)){
21             return new BContent();
22         }
23
24         return new Content();
25     }
26
27 }
28
```

#퀴즈

메인화면에서 로그인 화면으로 이동하는 "로그아웃"은 어떻게 하면 될까요?



OPEN CLOUD ENGINE



Parameter 기반 서비스 메서드

UI없이 서비스 메서드를 다른 클래스에서 그냥 호출하여 사용하고 싶은 경우 (모델 재사용 혹은 테스트 작성시 등)

eg.

```
Main main = Login.login("jjy", "password");  
System.out.println("menu name is correct? " + ""main.getMenu()....);
```

아래와 같이 선언을 변경함:

```
@ServiceMethod  
public static login(Payload("id") String name, Payload("password") String  
password)
```



@AutowiredFromClient 의 사용

```
public class Main{  
    @AutowiredFromClient  
        public Session session;  
  
    @ServiceMethod(inContextMenu=true, callByContent=true)  
        public Session logout(){  
            session.expire();  
            return new Login();  
        }  
}
```



@AutowiredFromClient 가 많이 쓰여진 경우의 객체 재사용

위의 클래스를 통째로 재사용하고 싶은 경우 (//@Autowired가 많은 경우 매우 효율적임)

Old mw3:

```
Main main = new Main();  
main.session = session;  
main.logout();
```

New mw3:

```
Main main = new Main();  
autowire(main);  
main.logout();
```



Parameter에 Autowired넣어사용

특정 서비스메서드에서만 클라이언트 객체를 같이 끌고 올라올 필요가 있는 경우

```
public class Main{  
    // @AutowiredFromClient  
    //      public Session session;  
  
    @ServiceMethod(inContextMenu=true)  
    public static Session logout(  
        @AutowiredFromClient Session session)  
    {  
        session.expire();  
        return new Login();  
    }  
}
```

➔ In use:

```
return Main.logout(session);    //재사용성이 매우 높아짐
```



이벤트 리스닝

- `@ServiceMethod(eventBinding="methodName", bindingFor="propertyName")`



모델객체와 UI객체의 분리

UI객체 매핑 방법

```
@Face(faceClass=CompetencySelectBox.class)
public class Competency extends LanguageElement{
    ...
}
```

#org.metaworks.Face

```
package org.metaworks;
```

```
public interface Face<T> {
    public void setValueToFace(T value);
    public T createValueFromFace();
}
```



모델객체와 UI객체의 분리

UI객체 예시

```
public class CompetencySelectBox extends SelectBox implements Face<Competency>{
```

```
    @Override  
    public void setValueToFace(Competency value) {  
        setSelectedValue(value.getName());  
    }
```

```
    @Override  
    public Competency createValueFromFace() {  
        return new Competency(getSelectedText());  
    }
```

```
    public CompetencySelectBox(){  
        ArrayList<String> options = new ArrayList<String>();  
        options.add("1");  
        options.add("2");  
        setOptionNames(options);  
        setOptionValues(options);  
    }
```

```
}
```



Annotations

표현/표시 관련

@Face

@Hidden

@Name

행위 관련

@ServiceMethod

@AutowiredFromClient

@AutowiredToClient

DB 관련

@Table

@Id

@ORMMapping

데이터 관련

@Range

@Validator

@NonEditable

기타

@Available

@Test



Annotation: 객체와 필드의 얼굴정보

```
@Face(  
    displayName="화면에 뿌릴 명칭",  
    ejsPath="템플릿 파일 위치",  
)
```

```
@Hidden(  
    when: '보이지 않을 때'  
)
```

```
@Available(  
    when: '보일 수 있는 때'  
)
```



Annotation: 객체와 필드의 얼굴정보 - 용례



@Face / @Hidden / @

7분 전

```
1
2 @Face(
3     ejsPathMappingByContext=
4     {
5         "{when: 'new', face: 'faces/org/uengine/codi/mw3/model/IWorkItem_edit.ejs'}",
6         "{when: 'edit', face: 'faces/org/uengine/codi/mw3/model/IWorkItem_edit.ejs'}",
7         "{when: 'view', face: 'faces/org/uengine/codi/mw3/model/IWorkItem_view.ejs'}",
8     }
9 )
10
11 public interface IWorkItem extends IDAO{
12
13     @Face(displayName="타이틀")
14     public String getTitle();
15     public void setTitle(String title);
16
17     @Hidden(when="edit")
18     public String getExtFile();
19     public void setExtFile(String extFile);
20
21     @ServiceMethod(inContextMenu=true, needToConfirm=true, callByContent=true /*TODO: later add except*/)
22     @Face(displayName="$Delete")
23     @Available(when="view")
24     public Object remove() throws Exception;
25
26
27 }
28
29
```

👍 좋아요

Annotation: ejs 템플릿



ejs 파일의 형식

5분 전

```
1 <%
2     if(value){
3     %>
4
5     <table width="100%" border="0" cellspacing="0" cellpadding="0">
6         <tr>
7
8             <td width=16>&nbsp;
9             </td>
10
11             <td width=30>
12                 <%=fields.title.here()%>
13                 <%=fields.extFile.here()%>
14             </td>
15
16             <td><%=methods.remove.here()%></td>
17         </tr>
18
19     </table>
```

좋아요

* EJS: Embedded Javascript Template Engine 로, 기존 JSP, PHP 문법을 지원하는 클라이언트 기반 템플릿엔진



Annotation: 서비스 메서드

```
@ServiceMethod(  
    target="self | popup | auto",  
    callByContent=true|false,  
    payload={'field1', 'field2',...},  
    except={'field1', 'field2',...},  
    keyBinding="Alt|Shift|Ctrl+Key",  
    mouseBinding="left|right|dblclick",  
    inContextMenu=true|false,  
    needToConfirm=true|false,  
    when="context"  
)
```

```
//리턴값을 어디로?  
//속성값을 태울것인가?  
//요 속성들만 태워라  
//요 속성들은 빼고 태워라  
//키가 눌러지면 콜  
//마우스가 눌러지면 콜  
//컨텍스트 메뉴에 걸어  
//정말 할건지 물어  
//언제 활성화 될건지
```

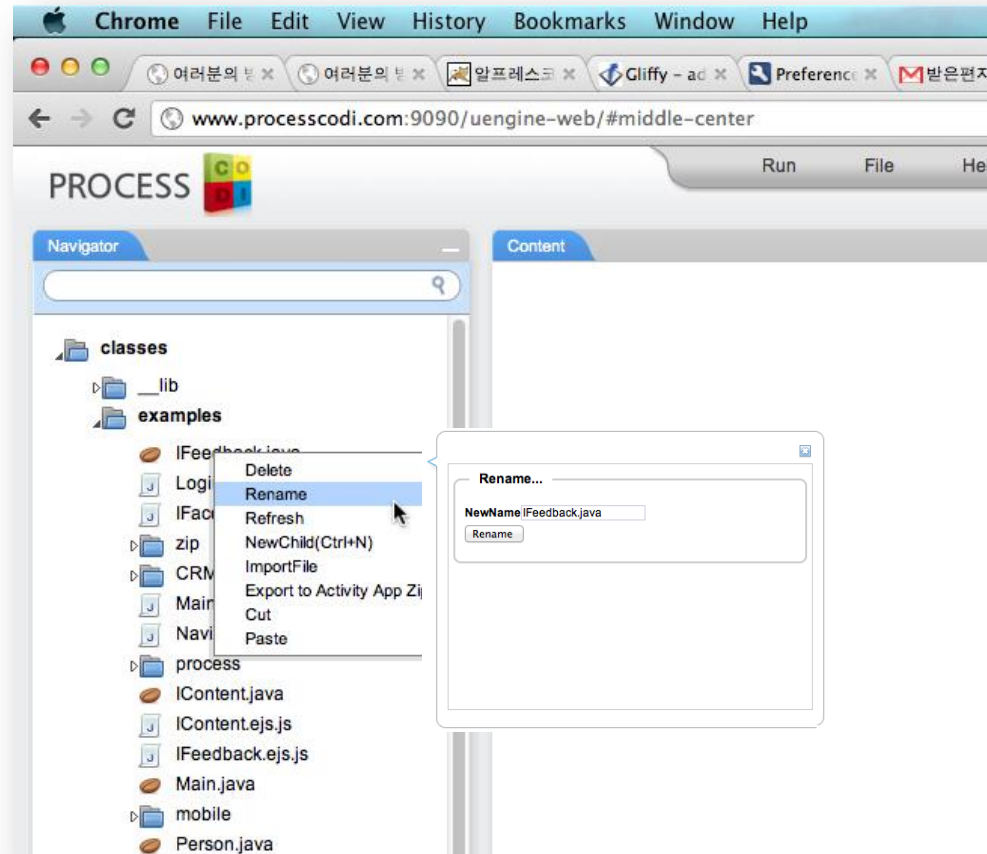


Annotation: 서비스 메서드

```
@ServiceMethod(inContextMenu=true, callByContent=true, target="popup", keyBinding="Ctrl+R")
public Popup rename(){
    FileRenamer fileRenamer = new FileRenamer();
    fileRenamer.setFile(this);

    Popup renamer = new Popup(fileRenamer);

    return renamer;
}
```



Annotation: 저장관련 정보 (JPA-호환)

<선언 관련>

```
@Table(  
    name="DB 테이블 명"  
)  
@Id // 프라이머리 키  
@NonSavable // 저장 필드 아님  
@NonLoadable // 읽을 필드 아님
```

<ORMapping 관련>

```
@TypeSelector(  
    values={'값1', '값2', ...},  
    classes={'서브클래스명1', '서브클래스명1',...}  
)  
@ORMapping(  
    objectFields={'객체속성1', '객체속성2', ...},  
    databaseFields={'테이블컬럼1', '테이블컬럼2',...}  
)
```



Annotation: 저장관련 정보 (JPA-호환)

Database 관련 Annotation 사용예: @Table / @Id / @TypeSelector / @ORMMapping

5분 전

```
1
2 @Table(name = "bpm_worklist")
3 public interface IWorkItem extends IDAO{
4
5     @Id
6     public Long getTaskId();
7     public void setTaskId(Long taskId);
8
9     @Hidden
10    @TypeSelector(values = { "comment", "img", "memo"}, classes = { CommentWorkItem.class, ImageWorkItem.class,
11    public String getType();
12    public void setType(String type);
13
14    @Hidden
15    public String getContent();
16    public void setContent(String content);
17
18    @ORMMapping(databaseFields = { "content" }, objectFields = { "contents" })
19    public WebEditor getMemo();
20    public void setMemo(WebEditor memo);
21
22
```

👍 좋아요



OPEN CLOUD ENGINE

Libraries

리턴 객체 위치잡기

ToPrepend

ToAppend

ToNext

팝업 관련

Popup

ModalWindow

ToOpener

레이아웃 잡기

Layout

교체하기/제거하기

Refresh

Remover

브로드캐스팅 하기

pushClientObjects

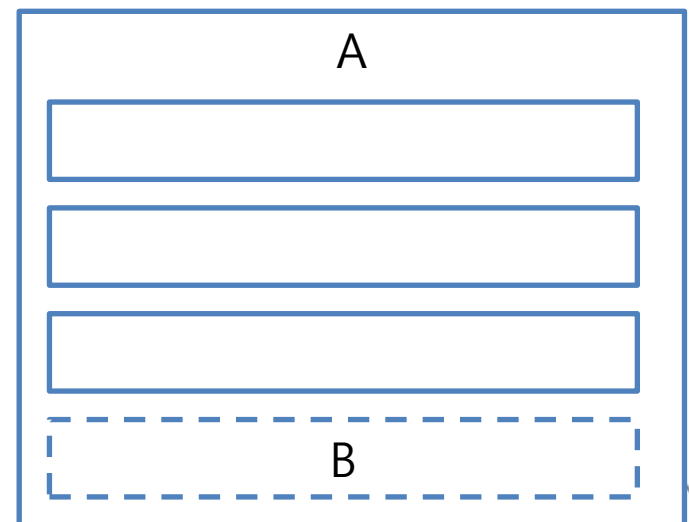
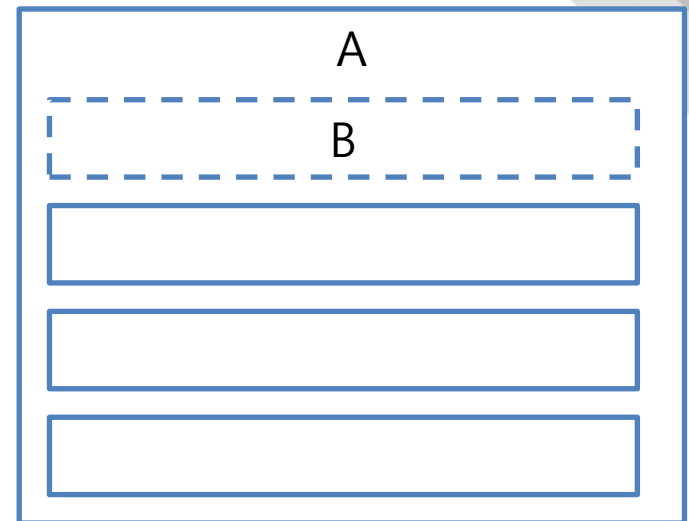
기본 가젯들

Grid
Window
Chart

리턴객체의 위치 잡기

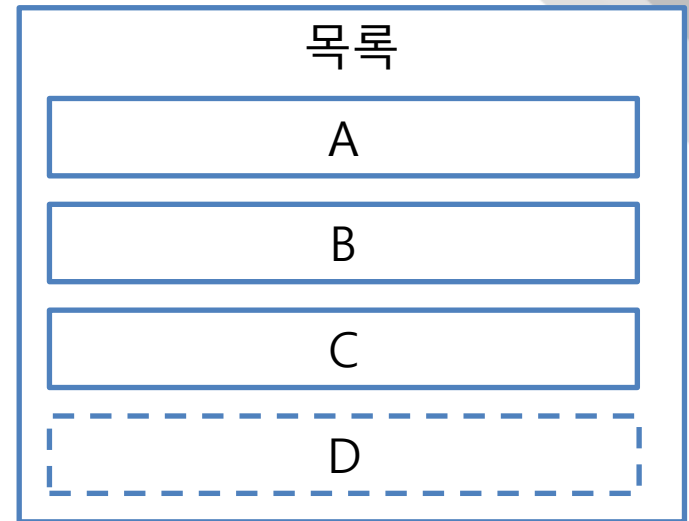
```
return new ToPrepend(  
    A, //추가될 영역을 가진 객체  
    B//추가할 객체  
);
```

```
return new ToAppend(  
    A, //추가될 영역을 가진 객체  
    B//추가할 객체  
);
```



리턴객체의 위치 잡기

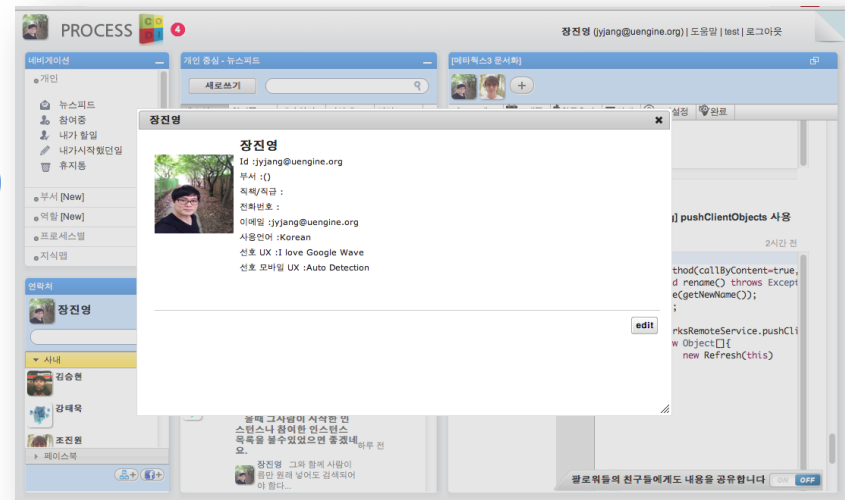
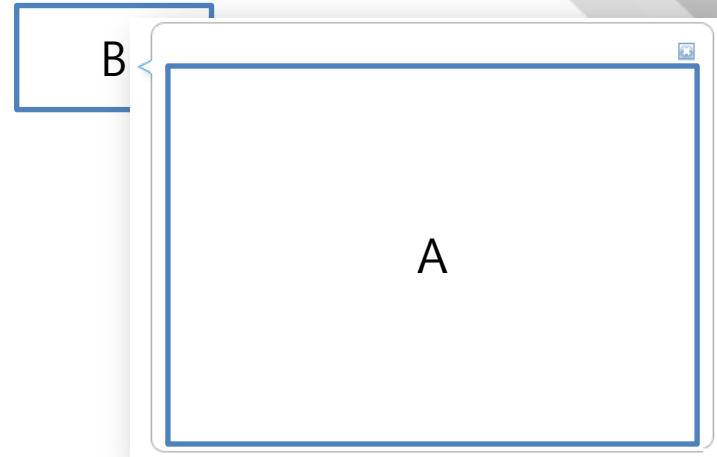
```
return new ToNext(  
    C, //추가될 영역을 가진 객체  
    D //추가할 객체  
);
```



팝업관련

```
public class B{  
    public Popup action(){  
        return new Popup(  
            A //팝업될 객체.  
        );  
    }  
}
```

```
return new ModalWindow(  
    A //팝업될 객체 (중앙배치)  
);
```





[Push 방식 BroadCasting] pushClientObjects 사용법

39분 전

```
1  
2 final Object[] returnObjects = new Object[]{  
3     new ToPrepend(new InstanceList(), refreshedInstance) // 인스턴스 리스트에 맨 꼭대기에  
4 };  
5  
6 MetaworksRemoteService.getInstance().pushClientObjects(returnObjects);  
7
```

ToPrepend 를 이용하여 객체를 포장
해서 브로드캐스트 했기 때문에 목록
을 보고 있는 유저에게만 추가된 아이
템이 나타남

사용자1: '오이'를 추가함

이름: 감자

이름:

오이

추가

사용자2: 목록을 보고 있는 유저

이름: 오이

이름: 고구마

사용자3: 디테일 화면에 들어가 있는 유저

이름: 고구마

고구마(학명: *Ipomoea batatas*)는 메꽃과의 한해살
이 뿌리 채소로, 주로 전분이 많고 단 맛이 나는 혹
뿌리를 가진 재배용 작물이다.

상황: 특정 아이템이나 주제에 대하여 화면상에 해당 아이템을 보고 있는 유저에게만 영향주기


[Push 방식 BroadCasting] pushClientObjects 사용법

```
1
2  @ServiceMethod(callByContent=true, keyBinding="enter")
3  public void rename() throws Exception{
4      setName(getNewName());
5      save();
6
7      MetaworksRemoteService.pushClientObjects(
8          new Object[]{
9              new Refresh(this)
10         }
11     );
12 }
```







Browser tabs: (4)Process Codi, view-source:www.processcodi.com, Google Project Hosting

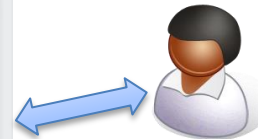
Address bar: www.processcodi.com:9090/uengine-web/#middle-center

PROCESS 

Knowledge Map

- 6월
 - SMART-TECH KOREA 2012
 - 
- 내부 교육
 - [394] 
 - 프로그래밍 교육
 - 1일자:
 - 컴퓨터 프로그래밍 언어란 무엇인가?
 - 컴퓨터를 8살 짜리에게 설명한다면?
 - 자연어와 프로그래밍 언어와의 해석의 시작 - UML (Unified Modeling Language)
 - 1일자 미션: 페이스북의 구조와 행위를 UML로 표현하시오.
 - [481]  안병성
 - [482]  김영재
 - 2일자:
 - Class 설계 실습
 - 모델링한 클래스를 문자열로 표현하기
 - 그냥 시키는대로 해야할 것들
 - 그 외 기본 상식
 - 해의(해서드) 내부의 면려는 무엇으로 채우는가?

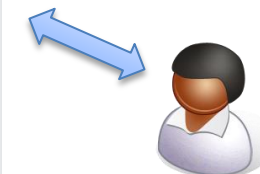
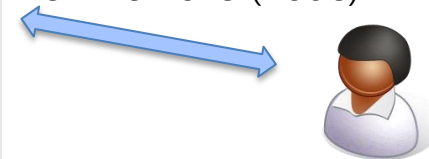
추가:
new ToNext(node)



수정:
new Refresh(node)



삭제:
new Remover(node)



OPEN CLOUD ENGINE

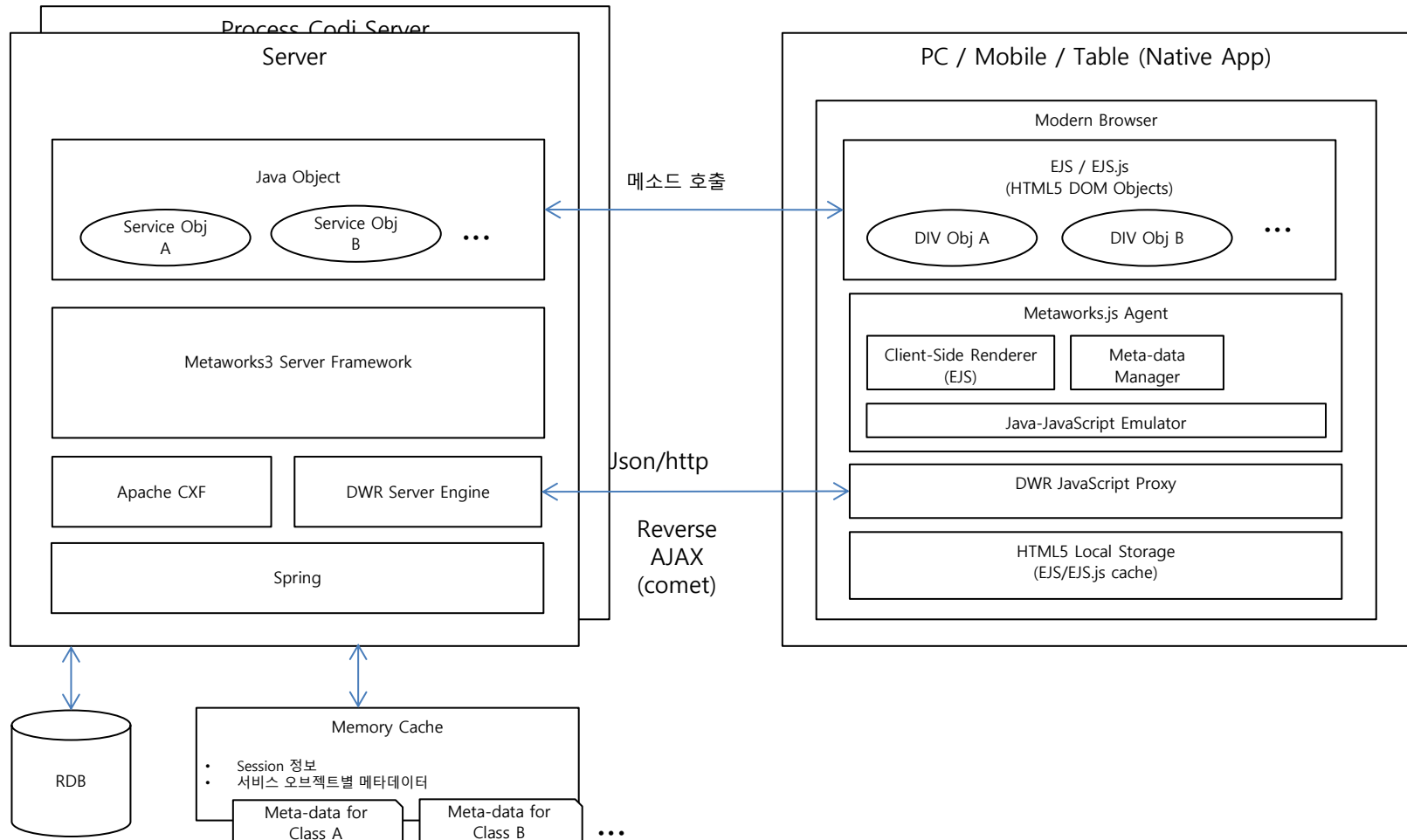
Anatomy of Metaworks



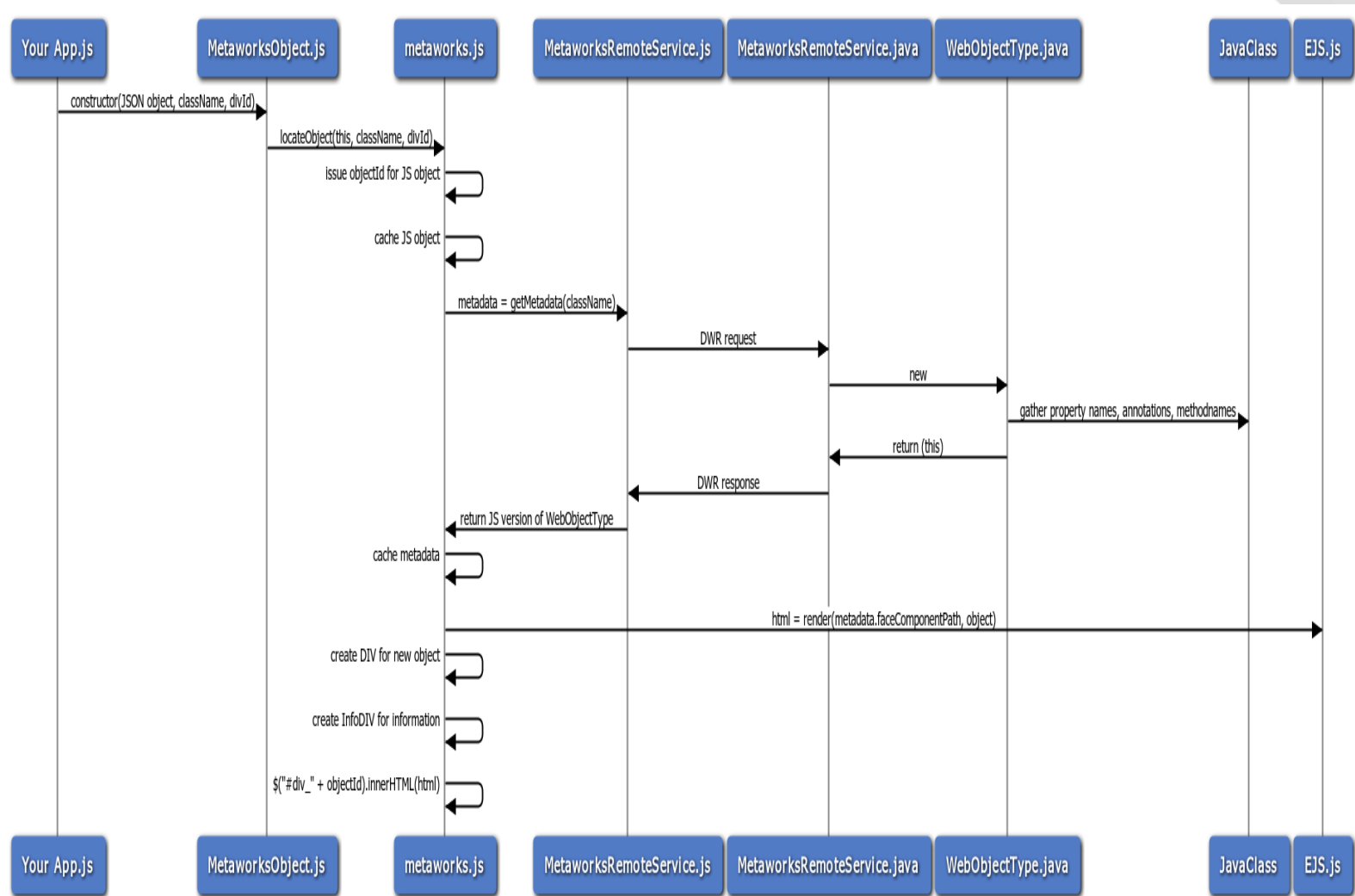
OPEN CLOUD ENGINE

Overview

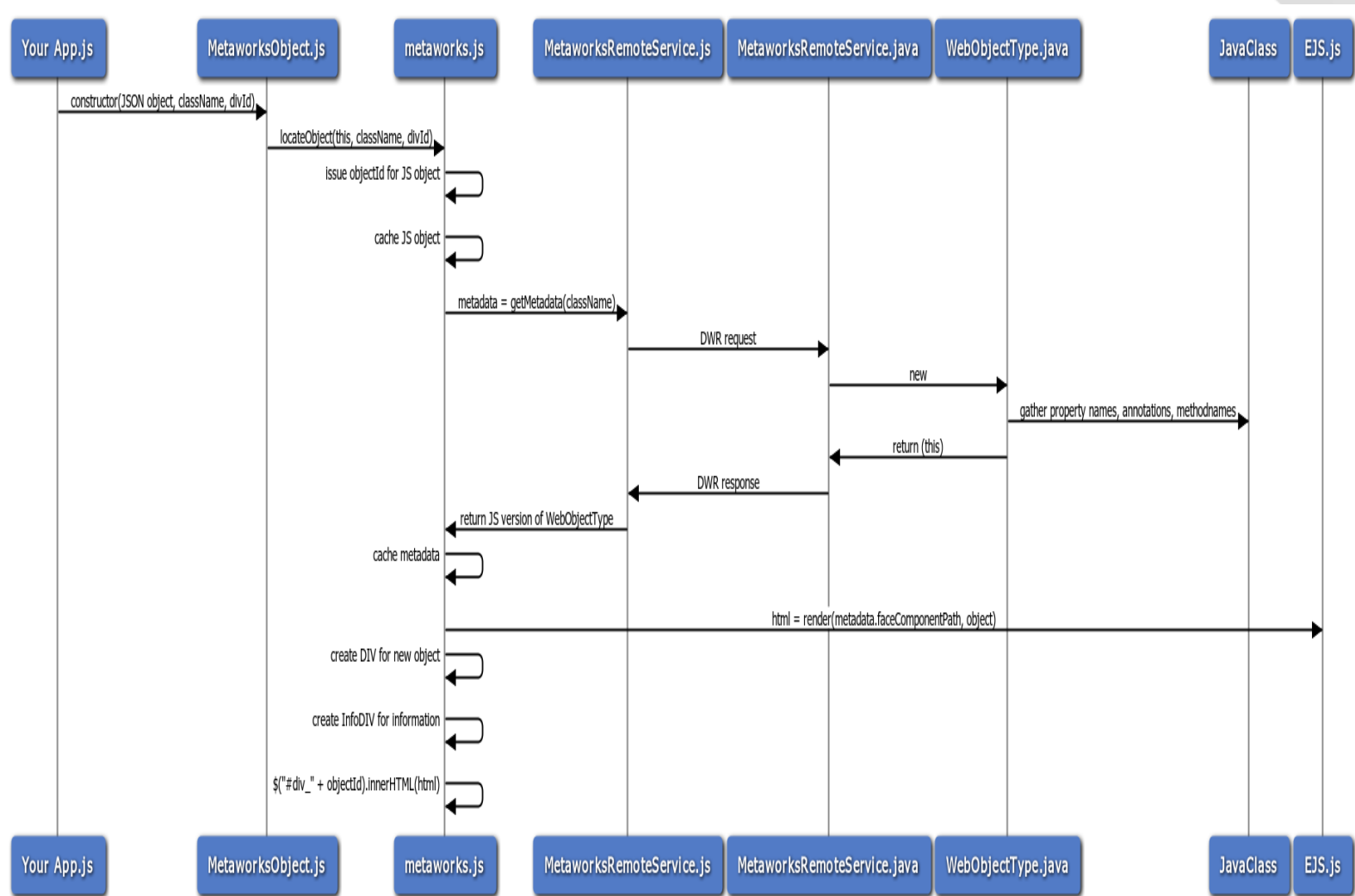
- Client-Side Rendering and JSON via HTTP
- Client Object Controlling through Reverse AJAX
- Abstract Object-Oriented Programming



Rendering an object



Method Invocation



Appendix.



OPEN CLOUD ENGINE

개발 관련 FAQ

1. 서버 관련 FAQ - 1

질문) 필드명과 메서드명을 동일하게 작성했더니 실행이 되지 않습니다.

```
int indent = 0;
@ServiceMethod
public void indent(){
    indent += 10;
}
```

답변)

서비스 메서드와 필드명이 동일한 경우 메타웍스3에서는 정상적으로 필드와 메서드를 구분하지 못해 오류를 발생시킵니다. 위의 코드는 다음과 같이 서로 다른 이름으로 사용하세요.

```
int indentDepth = 0;
@ServiceMethod
public void indent(){
    indentDepth += 10;
}
```

개발 관련 FAQ

1. 서버 관련 FAQ - 3

질문) 다음과 같이 코드를 하였는데 화면이 멈추어 버립니다.

```
class Node {  
    Node[] child;  
}
```

답변)

메타웍스3에서는 자신과 동일한 객체를 직접 가지면 재귀호출이 발생하게 됩니다. 따라서 자신과 동일한 객체를 소유하려면 ArrayList와 같은 Collection 등을 이용하여 우회적으로 사용하여야 합니다.

```
class Node {  
    ArrayList<Node> child;  
}
```

개발 관련 FAQ

1. 서버 관련 FAQ - 4

질문) 다음과 같이 서버에서 Annotation을 사용하여 Client 객체를 얻으려고 하는데 정상동작 하지 않습니다.

```
class MainPage {  
    @AutowiredFromClient  
    ClientSession session;  
}
```

답변)

메타웍스3에서는 클라이언트를 가져오는 객체에 대해서는 public 접근제한자에 국한해서 지원합니다.

```
class MainPage {  
    @AutowiredFromClient  
    public ClientSession session;  
}
```

개발 관련 FAQ

1. 서버 관련 FAQ - 5

질문) ServiceMethod Annotation을 사용했음에도 불구하고 has no method 'myMethod' 라는 오류가 발생합니다.

```
class MainPage implements IMainPage {  
    @ServiceMethod  
    public void myMethod() { ... }  
}
```

답변)

메타웍스3에서는 인터페이스를 상속하는 클래스에 대해서는 해당 인터페이스 메서드를 주 참조 클래스로 인식합니다. 따라서 구현되는 서비스 메서드의 원형을 인터페이스에서도 넣어 주어야 합니다.

```
Interface IMainPage {  
    @ServiceMethod  
    public void myMethod();  
}  
class MainPage implements IMainPage{  
    public void myMethod() { ... }  
}
```

개발 관련 FAQ

1. 서버 관련 FAQ - 6

질문) 자바 클래스를 정상적으로 생성했음에도 불구하고 No Converter for 'yourClass' 라는 오류가 발생합니다.

```
package my.one.two;  
class yourClass { ... }
```

답변)

메타웍스3에서는 사용하려는 패키지에 대해서는 dwr.xml 파일에 선언해 주어야 합니다. 이는 화면에서 참조하는 클래스를 인식하기 위함입니다.

dwr.xml

```
<convert converter="metaworks" match="my.one.two.*"/>
```

개발 관련 FAQ

1. 서버 관련 FAQ - 7

질문) ORMapping Annotation이 인식되지 않는것 같습니다.

```
@Table(name="Login")
interface ILogin extends IDAO {
    @Id
    @ORMapping(databaseFields={"id"}, objectFields={"num"})
    public int getNum();
    ....
    @ORMapping(databaseFields={"name"}, objectFields={"name"})
    public String getName();
    ....
}
```

답변)

ORMapping은 DB Table의 필드명과 자바 클래스의 필드명이 다를 경우에 인식이 됩니다. 따라서 동일한 이름으로 되어 있을 경우에는 ORMapping을 사용할 필요가 없습니다. 만약 사용하게 되면 어떤 공간을 저장 영역으로 써야 할 지에 대한 혼돈이 발생하게 됩니다.

개발 관련 FAQ

2. 클라이언트 관련 FAQ - 1

질문) ejs를 만들었는데 반영이 되지 않습니다. 이유가 무엇인가요?

답변)

ejs는 자바클래스와 동일한 폴더에 작성할 수도 있고, webapps 이하의 contents 영역에 자바클래스의 패키지와 똑같은 이름의 path 상에 작성할 수도 있습니다. 그러나 만약 두 군데 모두 ejs가 존재할 경우 자바클래스와 동일한 폴더 내에 작성된 ejs가 우선 적용됩니다.

만약 자바클래스에서 @Face Annotation을 통해 특정 ejs를 지정한 경우에는 @Face를 통해 지정된 ejs가 최우선 적용됩니다.

질문) ejs가 변경되었을 때 반영되는 속도가 느립니다. 빠르게 할 수 있나요?

답변)

ejs의 경로 URL을 웹 브라우저로 하나 열어 두고 변경 시 그곳을 refresh 시키면 ejs의 반영이 빨라 집니다.

개발 관련 FAQ

2. 클라이언트 관련 FAQ - 2

질문) 자바클래스로 일부의 값이 전달되어 오지 않습니다. 이유가 무엇인가요?

myEjs.ejs

ID : <%= fields.userId.here() %>

PW : <%= fields.password.here() %>

답변)

fields 객체를 사용할 경우 동일명의 필드를 호출하게 되면 가장 마지막에 입력된 값을 가져오게 됩니다. 위 예시와 같이 fields의 id 를 잘못하여 두번 사용하게 되면 실제 자바클래스에서는 id 값으로 pw 값이 들어오게 됩니다.

myEjs.ejs

ID : <%= fields.userId.here() %>

PW : <%= fields.password.here() %>

개발 관련 FAQ

2. 클라이언트 관련 FAQ - 3

질문) 화면이 뜨지 않고 깨집니다. 이유가 무엇인가요?

myEjs.ejs

```
<!-- <%= fields.xxx.here() %> -->
```

Hello!!!

답변)

자바클래스와 연동된 객체를 호출하는 것은 ejs의 코멘트에 의해 실행을 막을 수 없습니다. 결론적으로 ejs Template 엔진이 계속하여 코멘트 영역에 html을 넣게 되므로 화면이 깨지게 됩니다.

개발 관련 FAQ

3. 클라이언트 관련 스크립트 FAQ - 1

질문) ejs.js 생성자에서 자바클래스 객체를 얻어오지 못합니다. 이유는?

myEjs.ejs.js

```
var my_one_two_yourClass = function(objectId, className) {  
    var obj = mw3.getObject(objectId);  
}
```

답변)

ejs.js 내의 생성자와 getValue메서드 내에서는 mw3.getObject(objectId) 대신 mw3.object[objectId] 를 사용해야 합니다.

myEjs.ejs.js

```
var my_one_two_yourClass = function(objectId, className) {  
    var obj = mw3.object[objectId];  
}
```

개발 관련 FAQ

3. 클라이언트 관련 스크립트 FAQ - 2

질문) ejs.js 에서 자바클래스 객체의 필드를 얻고자 합니다.

myEjs.ejs.js

```
var my_one_two_yourClass = function(objectId, className) {  
    this.objectId = objectId;  
    this.className = className;  
    alert(this.userId); // ← 가능한가요?  
}
```

답변)

ejs.js 내의 this는 FaceHelper 클래스입니다. 따라서 자바클래스 객체를 획득 후 해당 필드를 얻으셔야 합니다.

myEjs.ejs.js

```
var my_one_two_yourClass = function(objectId, className) {  
    ....  
    var obj = mw3.objects[this.objectId];  
    alert(obj.userId);  
}
```

개발 관련 FAQ

3. 클라이언트 관련 스크립트 FAQ - 3

질문) 다음과 같은 오류는 왜 발생하나요?

Error marshalling data. See the logs for more details.

답변)

ejs.js 의 package_className.prototype.getValue 에서 값을 type 에 맞지 않게 넣을때 발생하므로 type 확인 및 넘겨주는 값(value) 확인하셔야 합니다.

질문) ej.js 파일이 반영이 되지 않습니다. 이유가 무엇인가요?

답변)

ejs.js는 기본적으로 ej.js 파일이 존재해야 그 경로를 통해 ej.js를 찾아내어 실행하게 됩니다. 따라서 ej.js 파일이 존재하는지 확인하세요.

Thank you



OPEN CLOUD ENGINE



OPEN CLOUD ENGINE