

Semi Project

# 클래스 설계 보고서

- 렌터카 사이트 -

---

빌려조

프로젝트명 : C&C 렌터카

팀장 : 이병준

조원 : 김한진 이아람  
홍동국 정욱재  
한윤희

# Contents


---

1. 공용 패키지
2. 회원 관련 패키지
3. 게시판 관련 패키지
4. P2P 관련 패키지
5. 예약 관련 패키지

---

# 공용 패키지

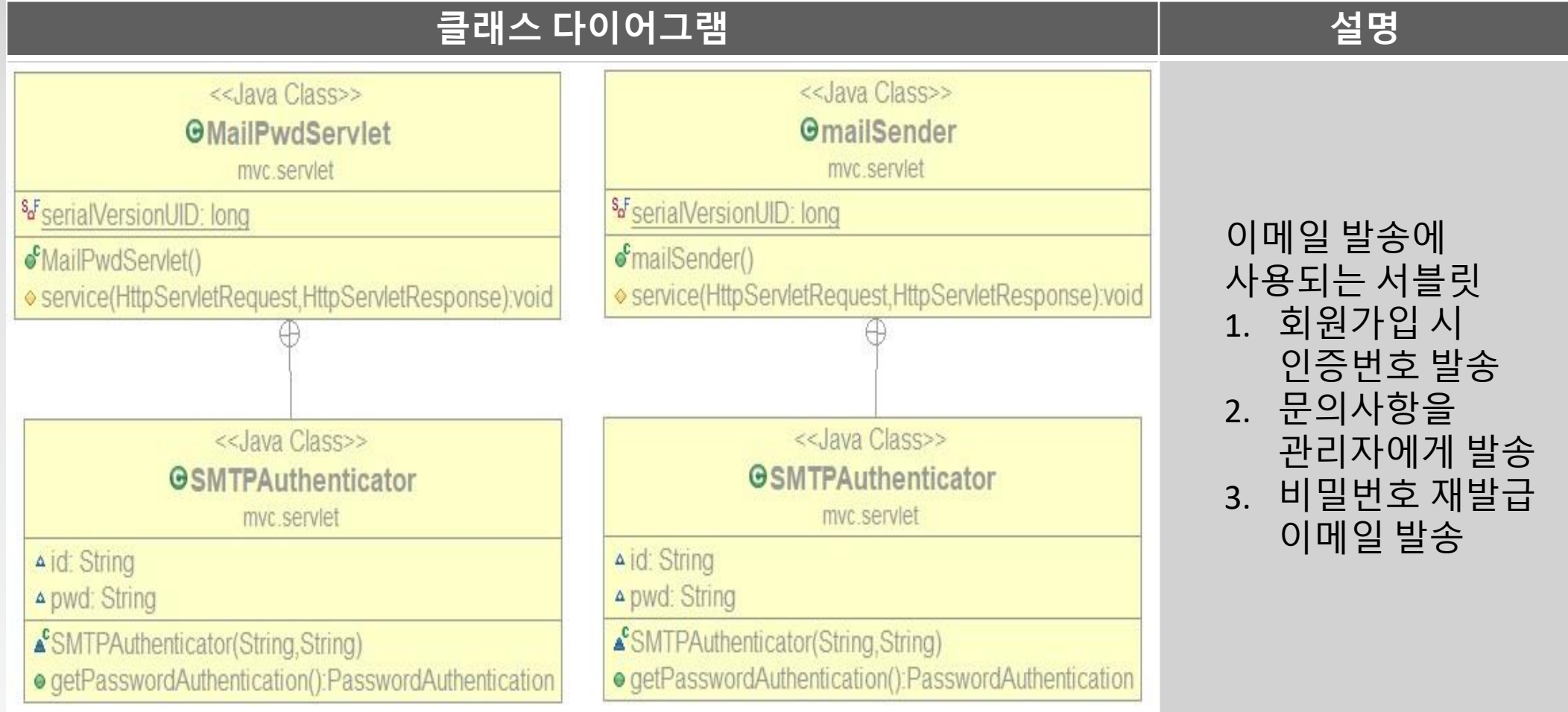
# 공용 패키지 - common

클래스 다이어그램	설명
 <pre>classDiagram     class JDBCTemplate {         &lt;&lt;Java Class&gt;&gt;         +JDBCTemplate()         +getConnection() Connection         +commit(Connection) void         +rollback(Connection) void         +close(Connection) void         +close(PreparedStatement) void         +close(Statement) void         +close(ResultSet) void     }     package common</pre> <p>The diagram shows a class named <b>JDBCTemplate</b> within the <b>common</b> package. It is a Java class. The methods listed are: <b>JDBCTemplate()</b> (constructor), <b>getConnection():Connection</b>, <b>commit(Connection):void</b>, <b>rollback(Connection):void</b>, <b>close(Connection):void</b>, <b>close(PreparedStatement):void</b>, <b>close(Statement):void</b>, and <b>close(ResultSet):void</b>. Each method is preceded by a green circle with a red 'c' for the constructor and a red 's' for static methods.</p>	<ol style="list-style-type: none"><li>1. 데이터베이스 연결용 클래스</li><li>2. 싱글톤 패턴 사용</li></ol>

# 공용 패키지 - Filter

클래스 다이어그램	설명
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>EncryptionFilter</b> filter</p></div> <div><ul style="list-style-type: none"><li>EncryptionFilter()</li><li>destroy():void</li><li>doFilter(ServletRequest,ServletResponse,FilterChain):void</li><li>init(FilterConfig):void</li></ul></div>	비밀번호 암호화용 필터 & 래퍼
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>EncryptionWrapper</b> wrapper</p></div> <div><ul style="list-style-type: none"><li>EncryptionWrapper(HttpServletRequest)</li><li>getParameter(String):String</li><li><u>getSha512(String):String</u></li></ul></div>	

# 공용 패키지 – mvc.servlet

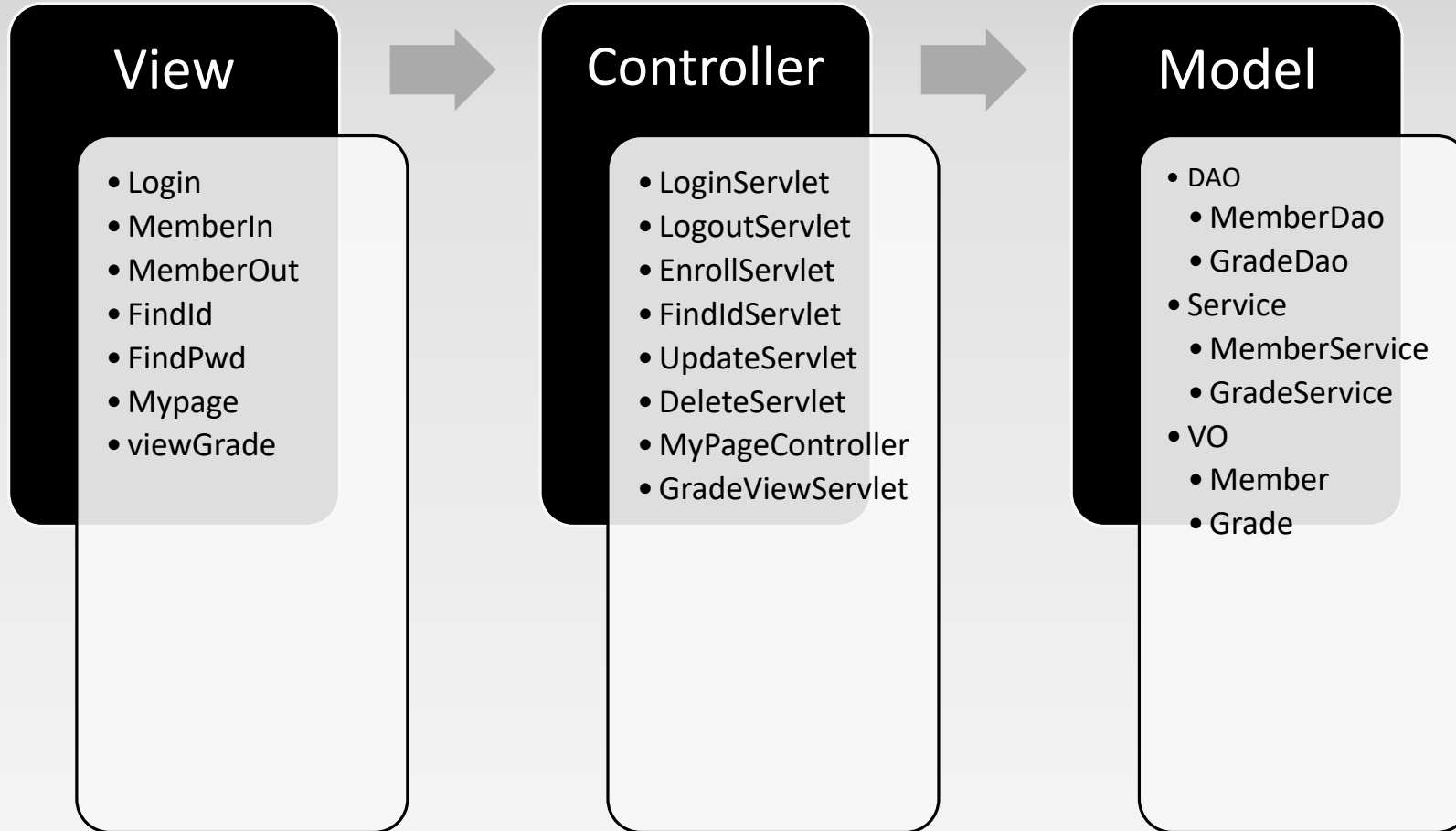


---

# 회원 관련 패키지

# MVC 패턴 - member

---







# 회원 관련 패키지 – Model - Service

클래스 다이어그램	설명
<p>&lt;&lt;Java Class&gt;&gt;</p> <p><b>MemberService</b></p> <p>member.model.service</p> <hr/> <p>MemberService()</p> <p>selectLogin(String, String): Member</p> <p>updatePwd(String, StringBuffer): int</p> <p>insertMember(Member): int</p> <p>selectFindId(String, String): String</p> <p>updateMemberPwd(String, String): int</p>	<p>회원 정보 관련 이벤트 처리를 위한 클래스</p> <p>ex) 로그인, 회원가입, 아이디찾기 등</p>

# 회원 관련 패키지 – Model - Service

클래스 다이어그램	설명
 <p>The diagram shows a Java class named <code>GradeService</code> with the package <code>grade.model.service</code>. It includes three methods: <code>GradeService()</code>, <code>selectList(): ArrayList&lt;Grade&gt;</code>, and <code>selectGrade(String): Grade</code>.</p>	마이페이지 등급 서비스용 클래스

# 회원 관련 패키지 – Model - Dao

클래스 다이어그램	설명
 <pre>classDiagram     class MemberDao {         &lt;&lt;Java Class&gt;&gt;         MemberDao()         selectLogin(Connection, String, String) Member         updatePwd(Connection, String, StringBuffer) int         insertMember(Connection, Member) int         selectFindId(Connection, String, String) String         updateMemberPwd(Connection, String, String) int     }</pre> <p>The diagram shows a class named <b>MemberDao</b> in the package <code>member.model.dao</code>. It has a constructor <code>MemberDao()</code> and six methods: <code>selectLogin(Connection, String, String): Member</code>, <code>updatePwd(Connection, String, StringBuffer): int</code>, <code>insertMember(Connection, Member): int</code>, <code>selectFindId(Connection, String, String): String</code>, and <code>updateMemberPwd(Connection, String, String): int</code>.</p>	<p>회원 정보 관련 데이터베이스 처리를 위한 Dao</p>

# 회원 관련 패키지 – Model - Dao

클래스 다이어그램	설명
 <pre>classDiagram     class GradeDao {         &lt;&lt;Java Class&gt;&gt;         GradeDao()         selectList(Connection): ArrayList&lt;Grade&gt;         selectGrade(Connection, String): Grade     }     package grade.model.dao</pre> <p>The diagram shows a class named GradeDao in the package grade.model.dao. It has a constructor GradeDao() and two methods: selectList(Connection): ArrayList&lt;Grade&gt; and selectGrade(Connection, String): Grade.</p>	마이페이지 등급 관련 처리를 하기 위한 Dao

# 회원 관련 패키지 – Model - VO

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; classDiagram     class Member {         mem_num int         email String         password String         name String         birthday String         phone String         address String         count int         cansell String         g_code String         Member()         Member(int, String, String, String, String, String, String, String, int, String, String)         getMem_num() int         setMem_num(int) void         getEmail() String         setEmail(String) void         getPassword() String         setPassword(String) void         getName() String         setName(String) void         getBirthday() String         setBirthday(String) void         getPhone() String         setPhone(String) void         getAddress() String         setAddress(String) void         getCount() int         setCount(int) void         getCansell() String         setCansell(String) void         getG_Code() String         setG_Code(String) void         toString() String     }     package member.model.vo</pre>	회원 정보가 저장될 vo

# 회원 관련 패키지 – Model - VO

클래스 다이어그램	설명
 <pre>classDiagram     class Grade {         g_code: String         g_rank: String         g_rate: int         g_comment: String         Grade()         Grade(String, String, int, String)         getG_code(): String         setG_code(String): void         getG_rank(): String         setG_rank(String): void         getG_rate(): int         setG_rate(int): void         getG_comment(): String         setG_comment(String): void         toString(): String     }     package grade.model.vo</pre> <p>The diagram shows a Java Class named <b>Grade</b> located in the package <code>grade.model.vo</code>. It has four attributes: <code>g_code: String</code>, <code>g_rank: String</code>, <code>g_rate: int</code>, and <code>g_comment: String</code>. The methods include a default constructor <code>Grade()</code>, a parameterized constructor <code>Grade(String, String, int, String)</code>, and getter/setter methods for each attribute: <code>getG_code(): String</code>, <code>setG_code(String): void</code>, <code>getG_rank(): String</code>, <code>setG_rank(String): void</code>, <code>getG_rate(): int</code>, <code>setG_rate(int): void</code>, <code>getG_comment(): String</code>, and <code>setG_comment(String): void</code>. There is also a <code>toString(): String</code> method.</p>	<p>등급 정보가 저장될 vo</p>



# 회원 관련 패키지 – Controller






클래스 다이어그램		설명
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>LoginServlet</b> member.controller</p><p>serialVersionUID: long count: int</p><p>LoginServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</p></div>	<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>LogoutServlet</b> member.controller</p><p>serialVersionUID: long</p><p>LogoutServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</p></div>	<ol style="list-style-type: none"><li>로그인용</li><li>로그아웃</li><li>아이디찾기</li><li>회원가입</li></ol>
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>FindIdServlet</b> member.controller</p><p>serialVersionUID: long</p><p>FindIdServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</p></div>	<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>EnrollServlet</b> member.controller</p><p>serialVersionUID: long</p><p>EnrollServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</p></div>	

# 회원 관련 패키지 – Controller

클래스 다이어그램	설명
<p>The diagram shows two Java classes, <b>UpdateServlet</b> and <b>DeleteServlet</b>, both located in the <code>member.controller</code> package. Both classes have a static final attribute <code>serialVersionUID: long</code>. <b>UpdateServlet</b> has a constructor <code>UpdateServlet()</code> and two methods: <code>doGet(HttpServletRequest, HttpServletResponse): void</code> and <code>doPost(HttpServletRequest, HttpServletResponse): void</code>. <b>DeleteServlet</b> also has a constructor <code>DeleteServlet()</code> and two methods: <code>doGet(HttpServletRequest, HttpServletResponse): void</code> and <code>doPost(HttpServletRequest, HttpServletResponse): void</code>.</p>	<ol style="list-style-type: none"><li>1. 회원정보 수정용</li><li>2. 회원정보 삭제용</li></ol>



# 회원 관련 패키지 – Controller

클래스 다이어그램	설명
<p>&lt;&lt;Java Class&gt;&gt;</p> <p> <b>MyPageController</b> member.controller</p> <hr/> <p> <u>serialVersionUID: long</u></p> <hr/> <p> MyPageController()</p> <p> doGet(HttpServletRequest, HttpServletResponse): void</p> <p> doPost(HttpServletRequest, HttpServletResponse): void</p> <hr/> <p>&lt;&lt;Java Class&gt;&gt;</p> <p> <b>GradeViewServlet</b> grade.controller</p> <hr/> <p> <u>serialVersionUID: long</u></p> <hr/> <p> GradeViewServlet()</p> <p> doGet(HttpServletRequest, HttpServletResponse): void</p> <p> doPost(HttpServletRequest, HttpServletResponse): void</p>	<ol style="list-style-type: none"><li>1. 회원정보 조회용</li><li>2. 회원등급 조회용</li></ol>

# 회원 관련 패키지 – View

---

## Login.jsp

- 로그인, 회원가입, 아이디찾기, 비밀번호재설정 View

## Mypage.jsp

- 로그인 한 유저 자신의 정보를 볼 수 있는 View

## viewGrade.jsp

- 회원 등급 조회용 View

---

# 게시판 관련 패키지

# MVC 패턴 - board

## View

- reviewRent
- reviewRentDetail
- reviewRentUpdate
- reviewRentWrite
- FreeBulletinBoard
- FreeBulletinBoardDetail
- FreeBulletinBoardUpdate
- FreeBulletinBoardWrite
- Faq
- noticeAdminWrite
- noticeDetailView
- noticeError
- noticeList
- noticeUpdate
- report

## Controller

- FaqListServlet
- FaqSearchListServ
- NoticeDeleteServl
- NoticeUpdateSer
- NoticeDetailServl
- NoticeListServlet
- NoticeAdmin
- ReportListServlet
- ReportDeleteServ
- ReportUpdateSer
- ReportDetailServ
- ReportFileDownS
- ReviewRentWrite
- ReviewRentUpdat
- ReviewRentDelet
- ReviewRentDetail
- FreeWrite
- FreeDelete


## Model

- DAO
  - ReviewRentDao
  - FreeBulletinBoar  
dDao
  - FaqDao
  - NoticeDao
  - ReportDao
- Service
  - ReviewRentServi  
ce
  - FreeBulletinBoar  
dService
  - FaqService
  - NoticeService
  - ReportService
- VO
  - Faq, notice,report
  - Review,Free

---

# 게시판 - FAQ

# 게시판 관련 패키지 – Model - Service

클래스 다이어그램	설명
 <pre>classDiagram     class FaqService {         &lt;&lt;Java Class&gt;&gt;         FaqService()         getListCount() int         selectList(int, int) ArrayList&lt;Faq&gt;         selectSearchList(String) ArrayList&lt;Faq&gt;     }</pre> <p>The diagram shows a class named <b>FaqService</b> with the package <code>faq.model.service</code>. It includes four methods: a constructor <code>FaqService()</code>, <code>getListCount():int</code>, <code>selectList(int,int):ArrayList&lt;Faq&gt;</code>, and <code>selectSearchList(String):ArrayList&lt;Faq&gt;</code>.</p>	FAQ 게시판 처리를 담당하는 서비스 클래스

# 게시판 관련 패키지 – Model - Dao

클래스 다이어그램	설명
 <pre>classDiagram     class FAQDao {         &lt;&lt;Java Class&gt;&gt;         FAQDao()         getListCount(Connection) int         selectList(Connection, int, int) ArrayList&lt;Faq&gt;         selectSearchList(Connection, String) ArrayList&lt;Faq&gt;     }     package faq.model.dao</pre> <p>The diagram shows a class named <b>FAQDao</b> in the package <code>faq.model.dao</code>. It is a Java Class. The methods listed are: <code>FAQDao()</code>, <code>getListCount(Connection):int</code>, <code>selectList(Connection,int,int):ArrayList&lt;Faq&gt;</code>, and <code>selectSearchList(Connection,String):ArrayList&lt;Faq&gt;</code>.</p>	FAQ 게시판 처리를 담당하는 DAO 클래스

# 게시판 관련 패키지 – Model - VO

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; classDiagram     class Faq {         serialVersionUID: long         f_no: int         f_category: String         f_title: String         f_contents: String         Faq()         Faq(int, String, String, String)         getF_no(): int         setF_no(int): void         getF_category(): String         setF_category(String): void         getF_title(): String         setF_title(String): void         getF_contents(): String         setF_contents(String): void         getSerialVersionUID(): long         toString(): String     }     package faq.model.vo</pre>	FAQ 데이터를 저장할 vo 클래스



# 게시판 관련 패키지 – Controller

클래스 다이어그램	설명
<pre>classDiagram     class FaqListServlet {         &lt;&lt;Java Class&gt;&gt;         serialVersionUID: long         FaqListServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class FaqSearchListServlet {         &lt;&lt;Java Class&gt;&gt;         serialVersionUID: long         FaqSearchListServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }</pre> <p>The diagram shows two classes, FaqListServlet and FaqSearchListServlet, both in the faq.controller package. FaqListServlet has a serialVersionUID, a constructor, and two methods: doGet and doPost. FaqSearchListServlet also has a serialVersionUID, a constructor, and two methods: doGet and doPost.</p>	<p>FAQ 게시판이 로딩될 때 데이터베이스에서 정보를 가져와 화면에 뿌려주는 역할을 할 서블릿</p> <p>FAQ 게시판에서 카테고리 별 검색을 할 때 사용할 서블릿</p>


---

# 게시판 - Notice

# 게시판 관련 패키지 – Model - Service

클래스 다이어그램	설명
<div><p><b>&lt;&lt;Java Class&gt;&gt;</b></p><p><b>NoticeService</b></p><p>notice.model.service</p></div> <div><ul style="list-style-type: none"><li>NoticeService()</li><li>getListCount():int</li><li>selectList(int,int):ArrayList&lt;Notice&gt;</li><li>insertNotice(Notice):int</li><li>selectNotice(int):Notice</li><li>addReadCount(int):void</li><li>deleteNotice(int):int</li><li>updateNotice(Notice):int</li></ul></div>	<p>Notice Controller에서 요청받은 내용을 전달받아 처리하고 DAO로 넘겨주고, 다시 리턴받아 Controller로 넘겨주는 NoticeService클래스</p>

# 게시판 관련 패키지 – Model - Dao

클래스 다이어그램	설명
<div data-bbox="211 472 1225 696"><p>&lt;&lt;Java Class&gt;&gt;</p><p> <b>NoticeDao</b></p><p>notice.model.dao</p></div> <div data-bbox="211 711 1225 1286"><ul style="list-style-type: none"><li>● NoticeDao()</li><li>● getListCount(Connection):int</li><li>● selectList(Connection,int,int):ArrayList&lt;Notice&gt;</li><li>● insertNotice(Connection,Notice):int</li><li>● selectNotice(Connection,int):Notice</li><li>● updateNotice(Connection,Notice):int</li><li>● deleteNotice(Connection,int):int</li><li>● addReadCount(Connection,int):int</li></ul></div>	<p>Notice게시판 관련하여 DB에 직접 접근하여 게시판 글의 처리를 실질적으로 담당하는 NoticeDAO클래스</p>

# 게시판 관련 패키지 – Model - VO

## 클래스 다이어그램

## 설명

```
<<Java Class>>
classDiagram
    class Notice {
        serialVersionUID: long
        n_no: int
        n_title: String
        n_writer: String
        n_contents: String
        n_original_filename: String
        n_rename_filename: String
        n_sysdate: Date
        readCount: int
        Notice()
        Notice(int, String, String, String, String, String, String, Date, int)
        Notice(int, String, String, String, String)
        Notice(int, String, String)
        getN_no(): int
        setN_no(int): void
        getN_title(): String
        setN_title(String): void
        getN_writer(): String
        setN_writer(String): void
        getN_contents(): String
        setN_contents(String): void
        getN_original_filename(): String
        setN_original_filename(String): void
        getN_rename_filename(): String
        setN_rename_filename(String): void
        getN_sysdate(): Date
        setN_sysdate(Date): void
        getReadCount(): int
        setReadCount(int): void
    }
    package notice.model.vo
```

Notce(공지사항) 데이터를 저장할 vo 클래스

# 게시판 관련 패키지 – Controller(1)

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; classDiagram     class NoticeAdminWrite {         serialVersionUID: long         NoticeAdminWrite()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class NoticeDeleteServlet {         serialVersionUID: long         NoticeDeleteServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class NoticeDetailServlet {         serialVersionUID: long         NoticeDetailServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class NoticeListServlet {         serialVersionUID: long         NoticeListServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class NoticeUpdateReadyServlet {         serialVersionUID: long         NoticeUpdateReadyServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }</pre>	<p>관리자계정으로 로그인하였을때 실질적으로 공지를 작성할수 있는 NoticeAdminWriteServlet</p> <p>관리자계정으로 로그인하였을때 실질적으로 공지를 삭제할 수 있는 NoticeDeleteServlet</p> <p>회원 비회원 관리자 모두 공지사항 전체리스트페이지에서 원하는 글에대한 조회 처리용 NoticeDetailServlet</p> <p>공지사항 페이지로 이동하였을때 데이터베이스 Notice 테이블에 있는 모든정보를 끌어와 화면단에 뿌려주는 NoticeListServlet</p> <p>관리자계정으로 공지를 수정할때 해당공지에대한 정보에 대한 DB에 실질적인 데이터가 있는지 여부를 판단하여 결과처리하는 NoticeUpdateReadyServlet</p>

# 게시판 관련 패키지 – Controller(2)

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; c NoticeUpdateServlet   notice.controller  SF serialVersionUID: long  NoticeUpdateServlet() doGet(HttpServletRequest, HttpServletResponse):void doPost(HttpServletRequest, HttpServletResponse):void</pre>	<p>NoticeUpdateReadyServlet으로 수정을 시도한 공지 페이지의 실질적인 데이터가 있음을 확인한 이후 수정페이지로 넘어와 수정페이지에서 관리가 원하는 데이터로 수정을하여 값을 전달받고 처리하는 NoticeUpdateServlet</p>

---

# 게시판 - Report



# 게시판 관련 패키지 – Model - Service

클래스 다이어그램	설명
 <pre>classDiagram     class ReportService {         &lt;&lt;Java Class&gt;&gt;         selectList(int, int) ArrayList&lt;Report&gt;         getListCount() int         addReadCount(int) void         selectReport(int) Report         insertReport(Report) int         deleteReport(int) int         updateReport(Report) int     }</pre> <p>The diagram shows a class named <b>ReportService</b> with the package <code>report.model.service</code>. It lists the following methods:</p> <ul style="list-style-type: none"><li><code>ReportService()</code></li><li><code>selectList(int, int): ArrayList&lt;Report&gt;</code></li><li><code>getListCount(): int</code></li><li><code>addReadCount(int): void</code></li><li><code>selectReport(int): Report</code></li><li><code>insertReport(Report): int</code></li><li><code>deleteReport(int): int</code></li><li><code>updateReport(Report): int</code></li></ul>	<p>ReportController에서 요청받은 내용을 전달받아 처리하고 DAO로 넘겨주고, 다시 리턴받아 Controller로 넘겨주는 ReportService클래스</p>

# 게시판 관련 패키지 – Model - Dao

클래스 다이어그램	설명
<div data-bbox="198 454 1238 686"><p>&lt;&lt;Java Class&gt;&gt;</p><p> <b>ReportDao</b></p><p>report.model.dao</p></div> <div data-bbox="198 708 1238 1293"><ul style="list-style-type: none"><li> ReportDao()</li><li> selectList(Connection,int,int):ArrayList&lt;Report&gt;</li><li> getListCount(Connection):int</li><li> addReadCount(Connection,int):int</li><li> selectReport(Connection,int):Report</li><li> insertReport(Connection,Report):int</li><li> deleteReport(Connection,int):int</li><li> updateReport(Connection,Report):int</li></ul></div>	<p>Report게시판 관련하여 DB에 직접 접근하여 게시판 글의 처리를 실질적으로 담당하는 ReportDAO클래스</p>

# 게시판 관련 패키지 – Model - VO

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; classDiagram     class Report {         r_no: int         r_member_no: int         r_rank_no: String         r_title: String         r_writer: String         r_contents: String         r_original_filename: String         r_rename_filename: String         r_sysdate: Date         readCount: int         Report()         Report(int, int, String, String, String, String, String, String, Date, int)         Report(int, String, String)         getR_no(): int         setR_no(int): void         getR_member_no(): int         setR_member_no(int): void         getR_rank_no(): String         setR_rank_no(String): void         getR_title(): String         setR_title(String): void         getR_writer(): String         setR_writer(String): void         getR_contents(): String         setR_contents(String): void         getR_original_filename(): String         setR_original_filename(String): void         getR_rename_filename(): String         setR_rename_filename(String): void         getR_sysdate(): Date         setR_sysdate(Date): void         getReadCount(): int         setReadCount(int): void         toString(): String     }     package report.model.vo</pre>	Report(불량고객 신고)관련 데이터를 저장할 VO 클래스

# 게시판 관련 패키지 – Controller(1)

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; classDiagram     class ReportDeleteServlet {         serialVersionUID: long         ReportDeleteServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class ReportDetailServlet {         serialVersionUID: long         ReportDetailServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class ReportListServlet {         serialVersionUID: long         ReportListServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class ReportFileDownServlet {         serialVersionUID: long         ReportFileDownServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }</pre>	<p>유저가 로그인하였을때 Session조회하여 해당유저가 올린 글에 해당되는 것만 삭제처리하는 ReportDeleteServlet</p> <p>로그인한 유저에 관해서만 게시글의 상세조회페이지로 이동가능하게 하는 ReportDetailServlet</p> <p>불량고객신고 페이지로 이동하였을때 데이터베이스 Report 테이블에 있는 모든정보를 끌어와 화면단에 뿌려주는 ReportListServlet</p> <p>불량고객신고 상세조회 페이지에서 첨부파일이 존재할 때 첨부파일 다운로드 시행시 처리하는 ReportFileDownServlet</p>











# 게시판 관련 패키지 – Controller(2)

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; C ReportWriteServlet report.controller  SF serialVersionUID: long  ReportWriteServlet() doGet(HttpServletRequest, HttpServletResponse):void doPost(HttpServletRequest, HttpServletResponse):void</pre>	로그인된 사용자가 글 작성시 결과를 처리하는 ReportWriteServlet
<pre>&lt;&lt;Java Class&gt;&gt; C ReportUpdateReadyServlet report.controller  SF serialVersionUID: long  ReportUpdateReadyServlet() doGet(HttpServletRequest, HttpServletResponse):void doPost(HttpServletRequest, HttpServletResponse):void</pre>	로그인한 사용자가 작성한 글에대하여 수정을 시도할때 해당글의 해당되는 Report DB테이블로 접근하여 해당되는 자료가 있는지 조회를 하고 그 결과에따른 처리를 하는 ReportUpdateReadyServlet
<pre>&lt;&lt;Java Class&gt;&gt; C ReportUpdate report.controller  SF serialVersionUID: long  ReportUpdate() doGet(HttpServletRequest, HttpServletResponse):void doPost(HttpServletRequest, HttpServletResponse):void</pre>	ReportUpdateReadyServlet의 true값에 따른 View화면에서 사용자가 수정한 값의 DATA UPDATE를 처리하는 ReportUpdateServlet






---

# 게시판 - FreeBoard

# 자유게시판관련 패키지 – Model - Service

클래스 다이어그램	설명
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p> <b>FreeBoardService</b></p><p>freebulletinboard.model.service</p><hr/><ul style="list-style-type: none"><li> FreeBoardService()</li><li> selectList():List&lt;FreeBoard&gt;</li><li> selectFreeBoard(int):FreeBoard</li><li> insertFreeBoard(FreeBoard):int</li><li> updateFreeBoard(FreeBoard):int</li><li> deleteFreeBoard(int):int</li><li> selectSearchTitle(String):List&lt;FreeBoard&gt;</li><li> selectSearchDate(Date,Date):List&lt;FreeBoard&gt;</li><li> selectSearchWriter(String):List&lt;FreeBoard&gt;</li></ul></div>	<p>자유게시판 관련</p> <p>이벤트를 처리하는 클래스</p>

# 자유게시판관련 패키지 – Model - Dao

클래스 다이어그램	설명
<div>&lt;&lt;Java Class&gt;&gt;  <b>FreeBoardDao</b> freebulletinboard.model.dao</div> <div> FreeBoardDao()  selectList(Connection):List&lt;FreeBoard&gt;  selectFreeBoard(Connection,int):FreeBoard  insertFreeBoard(Connection,FreeBoard):int  updateFreeBoardDao(Connection,FreeBoard):int  deleteFreeBoard(Connection,int):int  selectSearchTitle(Connection,String):List&lt;FreeBoard&gt;  selectSearchDate(Connection,Date,Date):List&lt;FreeBoard&gt;  selectSearchWriter(Connection,String):List&lt;FreeBoard&gt;</div>	<p>자유게시판 처리를 담당하는 Dao 클래스</p>



# 자유게시판관련 패키지 – Model – VO

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; G FreeBoard freebulletinboard.model.vo  +fb_num: int +fb_subject: String +fb_writer: String +fb_content: String +fb_original_filename: String +fb_rename_filename: String +fb_hits: int +fb_date: Date +originalFilePath: String +renameFilePath: String  FreeBoard() FreeBoard(int,String,String,String,String,String,int,Date) getfb_num():int setfb_num(int):void getfb_subject():String setfb_subject(String):void getfb_writer():String setfb_writer(String):void getfb_content():String setfb_content(String):void getfb_original_filename():String setfb_original_filename(String):void getfb_rename_filename():String setfb_rename_filename(String):void getfb_hits():int setfb_hits(int):void getfb_date():Date setfb_date(Date):void toString():String getOriginalFilePath():String setOriginalFilePath(String):void getRenameFilePath():String setRenameFilePath(String):void</pre>	<p>자유게시판 데이터를 저장할 VO 클래스</p>

# 자유게시판관련 패키지 – Controller

클래스 다이어그램		설명
<div>1</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>FreeBoard</b></div> <div>freebulletinboard.controller</div> <div>SF serialVersionUID: long</div> <div> <div>FreeBoard()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	<div>2</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>FreeBoardDetail</b></div> <div>freebulletinboard.controller</div> <div>SF serialVersionUID: long</div> <div> <div>FreeBoardDetail()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	1. 자유게시판 글 목록  2. 자유게시판 상세 글  3. 자유게시판 글 작성  4. 자유게시판 글 수정 및 삭제
<div>3</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>FreeBoardWrite</b></div> <div>freebulletinboard.controller</div> <div>SF serialVersionUID: long</div> <div> <div>FreeBoardWrite()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	<div>4</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>FreeBoardUpdate</b></div> <div>freebulletinboard.controller</div> <div>SF serialVersionUID: long</div> <div> <div>FreeBoardUpdate()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	

# 자유게시판 관련 패키지 – View

---

## FreeBulletinBoard.jsp

- 자유게시판 게시글을 볼 수 있는 View

## FreeBulletinBoardDetail.jsp

- 자유게시판 글을 작성 할 수 있는 View

## FreeBulletinBoardUpdate.jsp

- 자유게시판 글을 작성자가 수정 및 삭제 할 수 있는 View






---

# 게시판 - RentReview

# 렌트후기 관련 패키지 – Model - Service

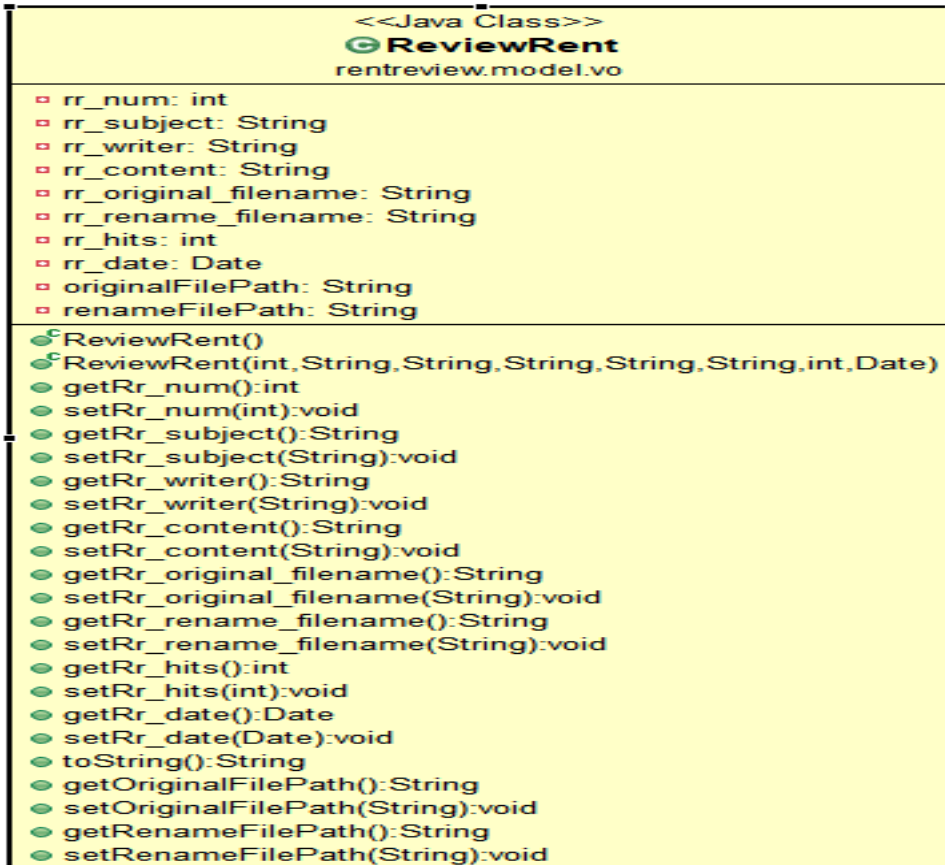
클래스 다이어그램	설명
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p><b>ReviewRentService</b></p><p>rentreview.model.service</p></div> <div><ul style="list-style-type: none"><li>ReviewRentService()</li><li>selectList():List&lt;ReviewRent&gt;</li><li>selectReviewRent(int):ReviewRent</li><li>insertReviewRent(ReviewRent):int</li><li>updateReviewRent(ReviewRent):int</li><li>deleteReviewRent(int):int</li><li>selectSearchTitle(String):List&lt;ReviewRent&gt;</li><li>selectSearchDate(Date,Date):List&lt;ReviewRent&gt;</li><li>selectSearchWriter(String):List&lt;ReviewRent&gt;</li></ul></div>	<p>렌트 후기 게시판 관련</p> <p>이벤트를 처리하는 클래스</p>

# 렌트후기 관련 패키지 – Model - Dao

클래스 다이어그램	설명
<div><p>&lt;&lt;Java Class&gt;&gt;</p><p> <b>ReviewRentDao</b></p><p>rentreview.model.dao</p></div> <div><ul style="list-style-type: none"><li> ReviewRentDao()</li><li> selectList(Connection):List&lt;ReviewRent&gt;</li><li> selectReviewRent(Connection,int):ReviewRent</li><li> insertReviewRent(Connection,ReviewRent):int</li><li> updateReviewRent(Connection,ReviewRent):int</li><li> deleteReviewRent(Connection,int):int</li><li> selectSearchTitle(Connection,String):List&lt;ReviewRent&gt;</li><li> selectSearchDate(Connection,Date,Date):List&lt;ReviewRent&gt;</li><li> selectSearchWriter(Connection,String):List&lt;ReviewRent&gt;</li></ul></div>	<p>렌트 후기 게시판 처리를</p> <p>담당하는 Dao 클래스</p>

# 렌트후기 관련 패키지 – Model - VO

## 클래스 다이어그램



## 설명

렌트 후기 게시판 데이터를

저장할 VO 클래스

# 렌트후기 관련 패키지 – Controller

클래스 다이어그램		설명
<div>1</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>ReviewRent</b></div> <div>rentreview.controller</div> <div>S_F serialVersionUID: long</div> <div> <div>ReviewRent()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	<div>2</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>ReviewRentDetail</b></div> <div>rentreview.controller</div> <div>S_F serialVersionUID: long</div> <div> <div>ReviewRentDetail()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	<div>1. 후기 글 목록</div> <div>2. 후기 상세 글</div> <div>3. 후기 글 작성</div> <div>4. 후기 글 수정 및 삭제</div>
<div>3</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>ReviewRentWrite</b></div> <div>rentreview.controller</div> <div>S_F serialVersionUID: long</div> <div> <div>ReviewRentWrite()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	<div>4</div> <div>&lt;&lt;Java Class&gt;&gt;</div> <div><b>ReviewRentUpdate</b></div> <div>rentreview.controller</div> <div>S_F serialVersionUID: long</div> <div> <div>ReviewRentUpdate()</div> <div>doGet(HttpServletRequest, HttpServletResponse):void</div> <div>doPost(HttpServletRequest, HttpServletResponse):void</div> </div>	



# 렌트후기 관련 패키지 – View

---

reviewRent.jsp

- 렌트후기 게시물 목록을 볼 수 있는 View

reviewRentDetail.jsp

- 실제 렌트 후기 글을 볼 수 있는 View

# 렌트후기 관련 패키지 – View

---

reviewRentWrite.jsp

- 렌트 이용자들이 후기 글을 작성 할 수 있는 View

reviewRentUpdate.jsp

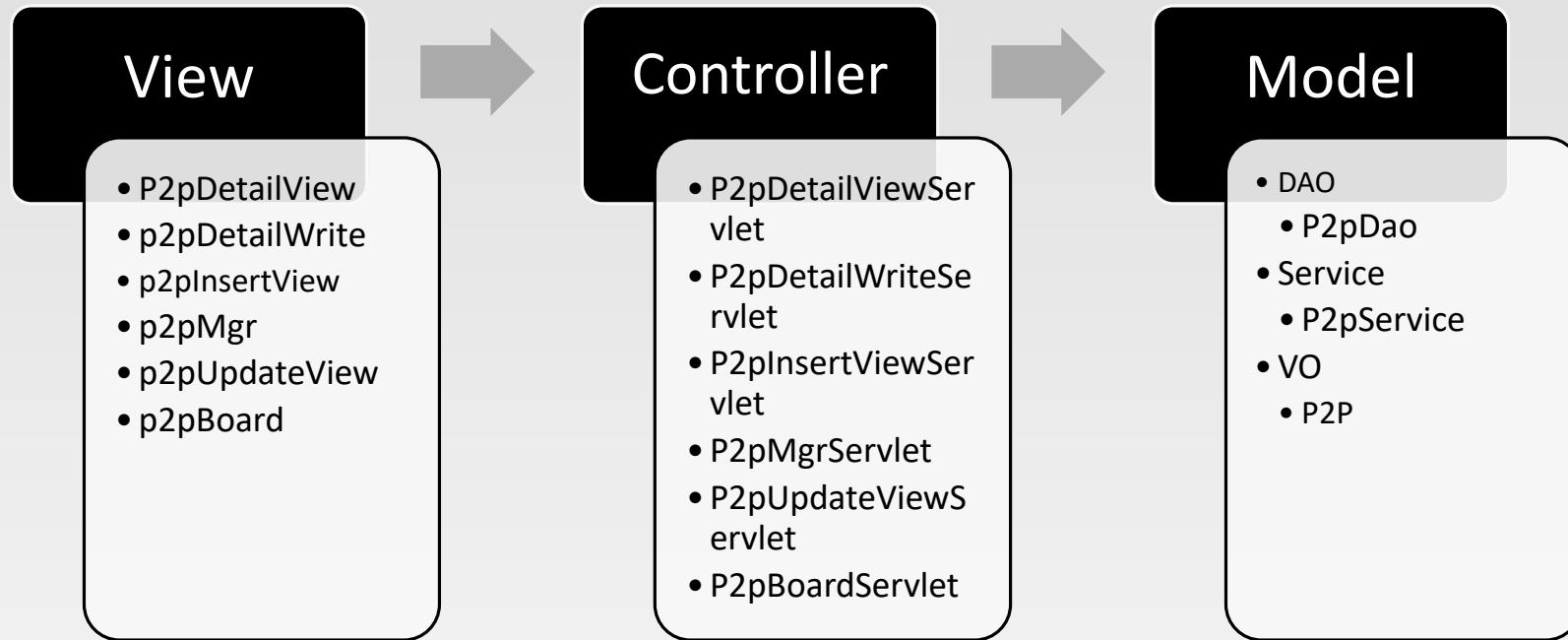
- 렌트 후기글을 작성자가 수정 및 삭제 할 수 있는 View

---


# P2P 관련 패키지

# MVC 패턴 – p2p


---




# P2P 관련 패키지 – Model - Service

클래스 다이어그램	설명
 <pre>classDiagram     class P2PService {         &lt;&lt;Java Class&gt;&gt;         P2PService()         getListCount() int         selectList(int, int) ArrayList&lt;P2P&gt;         insertBoard(P2P) int         selectBoard(int)         addReadCount(int) void         deleteBoard(int) int         updateBoard(P2P) int     }     package Aram.board.model.service</pre> <p>The diagram shows a class named <b>P2PService</b> within the package <b>Aram.board.model.service</b>. The class is a Java class and contains the following methods:</p> <ul style="list-style-type: none"><li><b>P2PService()</b> (constructor)</li><li><b>getListCount():int</b></li><li><b>selectList(int,int):ArrayList&lt;P2P&gt;</b></li><li><b>insertBoard(P2P):int</b></li><li><b>selectBoard(int)</b></li><li><b>addReadCount(int):void</b></li><li><b>deleteBoard(int):int</b></li><li><b>updateBoard(P2P):int</b></li></ul>	<p>P2p 관련 이벤트 처리를 위한 클래스</p>

# P2P 관련 패키지 – Model - Dao

클래스 다이어그램	설명
 <pre data-bbox="264 448 1322 1286">&lt;&lt;Java Class&gt;&gt; classDiagram     class P2PDao {         P2PDao()         insertP2P(HttpServletRequest) boolean         getProduct(String)         updateP2P(HttpServletRequest) boolean         reduceProduct(OrderBean) void         getP2PList() Vector         deleteP2P(String) boolean     }     package Aram.board.model.dao</pre> <p>The diagram shows a Java class named P2PDao located in the package Aram.board.model.dao. The class has the following methods:</p> <ul style="list-style-type: none"><li>P2PDao()</li><li>insertP2P(HttpServletRequest):boolean</li><li>getProduct(String)</li><li>updateP2P(HttpServletRequest):boolean</li><li>reduceProduct(OrderBean):void</li><li>getP2PList():Vector</li><li>deleteP2P(String):boolean</li></ul>	<p>P2p 관련 데이터베이스 처리를 위한 Dao</p>

# P2P 관련 패키지 – Model - VO

클래스 다이어그램	설명
 <pre>classDiagram     class p2p {         no: int         price: String         name: String         detail: String         date: String         image: String         p2p()         getNo(): int         setNo(int): void         getPrice(): String         setPrice(String): void         getName(): String         setName(String): void         getDetail(): String         setDetail(String): void         getDate(): String         setDate(String): void         getImage(): String         setImage(String): void     }</pre> <p>The diagram shows a Java Class named <code>p2p</code> with package <code>Aram.board.model.vo</code>. It contains six private attributes: <code>no</code> (int), <code>price</code> (String), <code>name</code> (String), <code>detail</code> (String), <code>date</code> (String), and <code>image</code> (String). The methods include a constructor <code>p2p()</code>, getters for all attributes, and setters for all attributes.</p>	<p>P2p 관련 데이터를 저장하기 위한 VO</p>

# P2P 관련 패키지 – Controller

클래스 다이어그램	설명
 <pre>classDiagram     class BoardDeleteServlet {         &lt;&lt;Java Class&gt;&gt;         serialVersionUID: long         BoardDeleteServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class BoardDetailServlet {         &lt;&lt;Java Class&gt;&gt;         serialVersionUID: long         BoardDetailServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class BoardFileDownServlet {         &lt;&lt;Java Class&gt;&gt;         serialVersionUID: long         BoardFileDownServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     BoardDeleteServlet -- &gt; BoardDetailServlet     BoardDeleteServlet -- &gt; BoardFileDownServlet</pre> <p>The diagram shows three Java classes: BoardDeleteServlet, BoardDetailServlet, and BoardFileDownServlet. Each class has a serialVersionUID attribute and implements doGet and doPost methods. BoardDeleteServlet is the superclass, and BoardDetailServlet and BoardFileDownServlet are subclasses.</p>	<p>P2p 게시물 삭제를 위한 서블릿</p> <p>P2p 게시물 상세보기하기 위한 서블릿</p> <p>P2P 게시판 파일 다운용 서블릿</p>



# P2P 관련 패키지 – Controller

클래스 다이어그램	설명
 <pre>classDiagram     class BoardOriginUpdateServlet {         serialVersionUID: long         BoardOriginUpdateServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class BoardListServlet {         serialVersionUID: long         BoardListServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }     class BoardUpdateViewServlet {         serialVersionUID: long         BoardUpdateViewServlet()         doGet(HttpServletRequest, HttpServletResponse): void         doPost(HttpServletRequest, HttpServletResponse): void     }</pre> <p>The diagram shows three Java Servlet classes: BoardOriginUpdateServlet, BoardListServlet, and BoardUpdateViewServlet. Each class has a serialVersionUID attribute and implements doGet and doPost methods. They are all part of the Aram.board.controller package.</p>	<p>P2P 게시판 파일 수정용 서블릿</p> <p>P2P 게시글 리스트 보기위한 서블릿</p> <p>P2P 게시글 수정용 서블릿</p>

# 회원 관련 패키지 – Controller

---

## P2PDetailViewServlet.java

- P2P 게시판 상세보기 처리용 서블릿

## P2PDetailWriteServlet.java

- P2P 게시판 작성 처리용 서블릿

## P2PInsertViewServlet.java

- P2p 게시판 삽입 처리용 서블릿

# 회원 관련 패키지 – Controller

---

## P2PMgrServlet.java

- P2P 게시판 관리 처리용 서블릿

## P2PUpdateViewServlet.java

- P2P 게시판 수정용 서블릿

## P2PBoardServlet.java

- P2P 게시판 뷰 처리용 서블릿

# 회원 관련 패키지 – View

---

## P2PDetailView.jsp

- P2p 시 상대가 올린 차량 정보를 상세하게 볼 수 있는 뷰

## P2PDetailWrite.jsp

- 상세 정보를 작성할 수 있게 해주는 뷰

## P2PInsertView.jsp

- P2p 에 글을 쓸 수 있는 폼을 제공해주는 뷰

# 회원 관련 패키지 – View

---

## P2PMgr.jsp

- P2p 시 상대가 올린 차량 정보를 상세하게 볼 수 있는 뷰

## P2PUpdateView.jsp

- P2p 정보를 수정할 수 있는 뷰

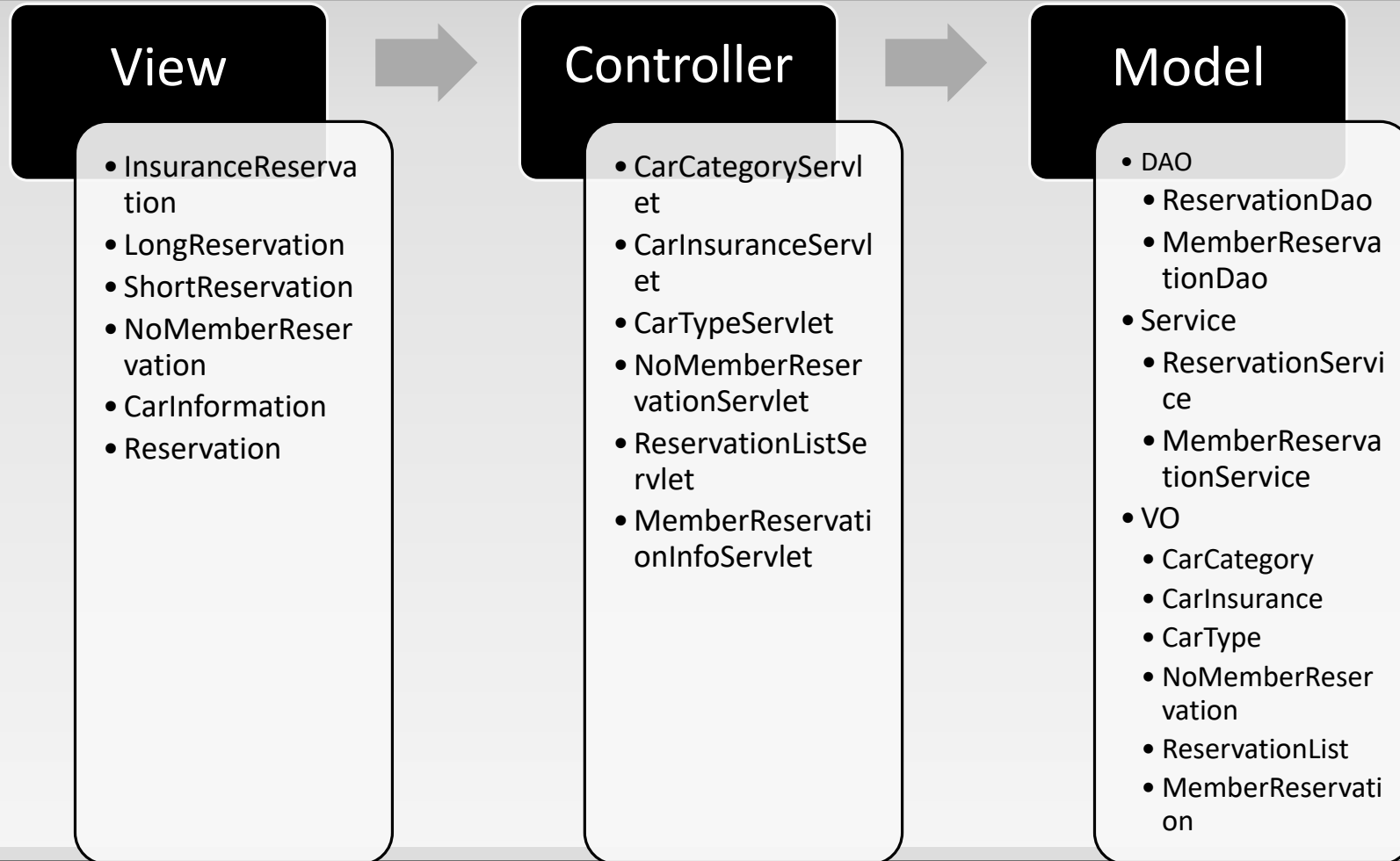
## P2PBoard.jsp

- P2p 메인 게시판을 보여주는 뷰

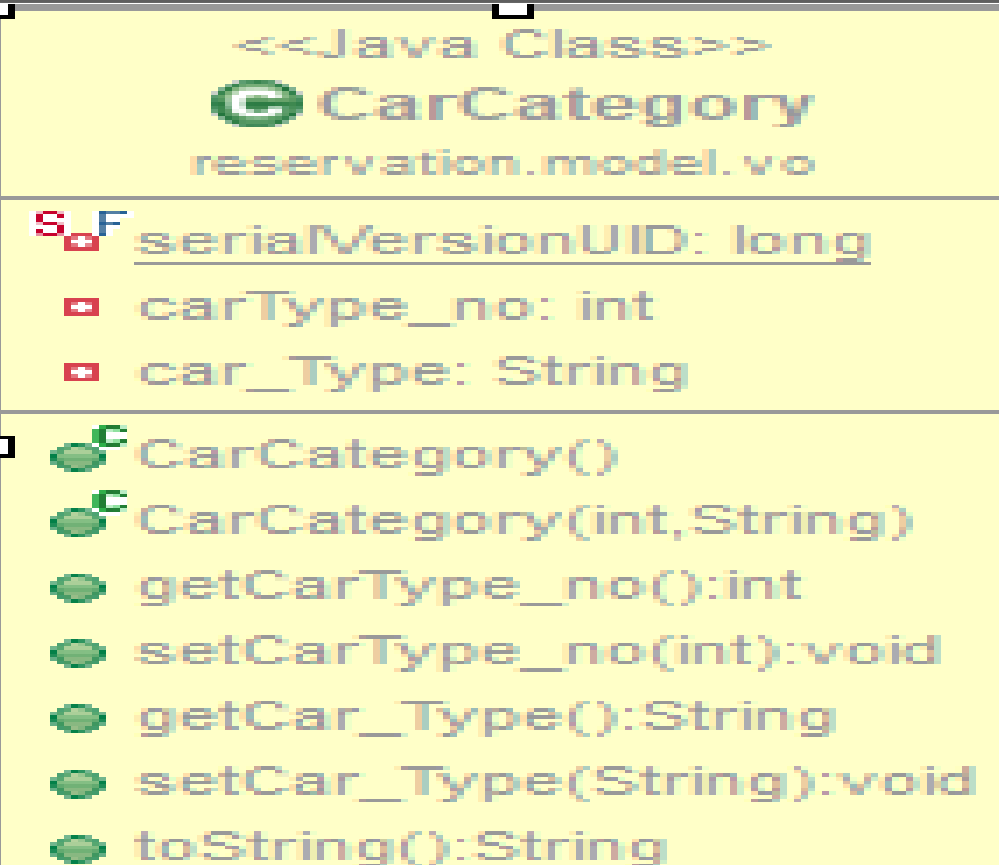
---

# 예약 관련 패키지

# MVC 패턴 – reservation

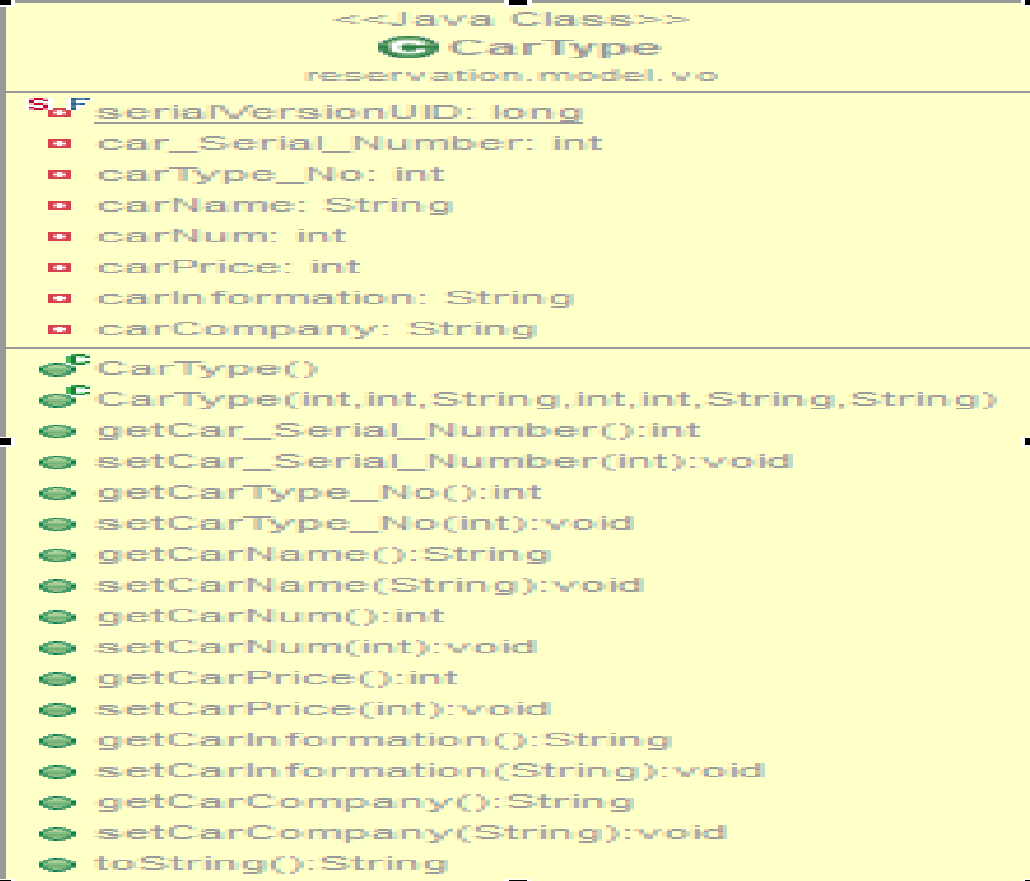


# 예약 관련 패키지 – Model -VO

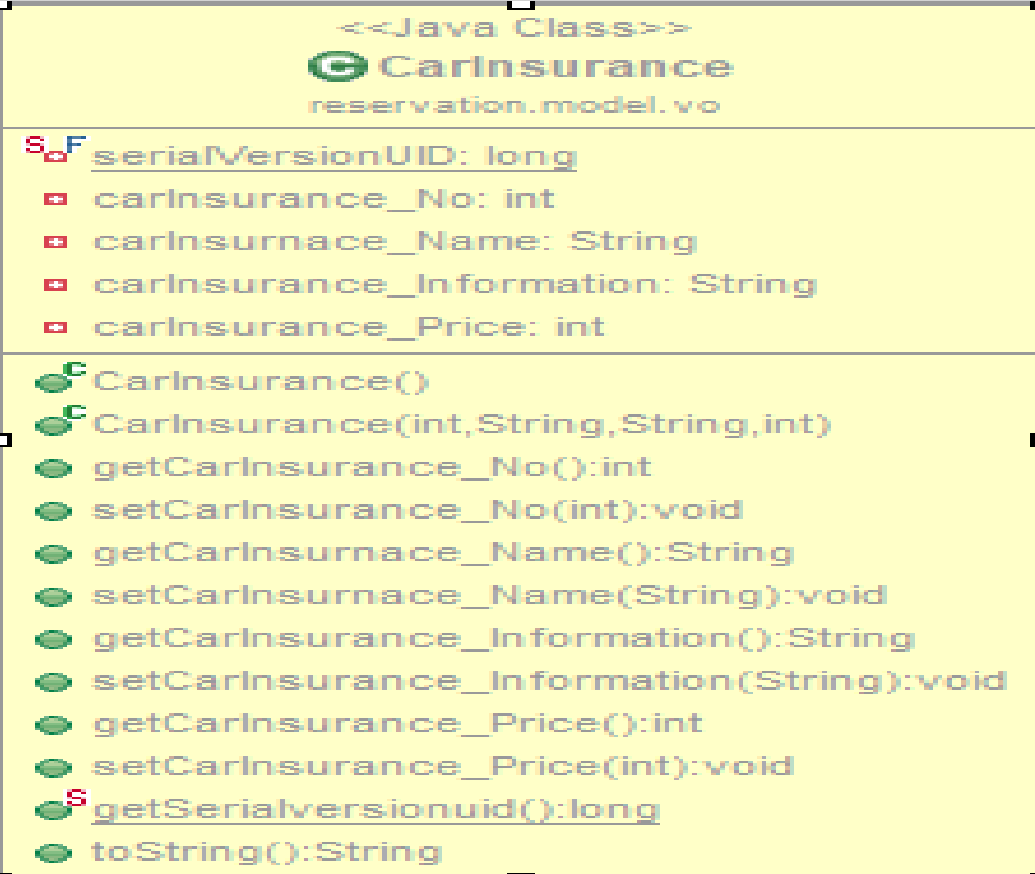
클래스 다이어그램	설명
 <pre>classDiagram     class CarCategory {         serialVersionUID: long         carType_no: int         car_Type: String         CarCategory()         CarCategory(int, String)         getCarType_no(): int         setCarType_no(int): void         getCar_Type(): String         setCar_Type(String): void         toString(): String     }</pre> <p>The diagram shows a class named <b>CarCategory</b> with the package <code>reservation.model.vo</code>. It has three attributes: <code>serialVersionUID: long</code> (final), <code>carType_no: int</code>, and <code>car_Type: String</code>. The methods include a no-argument constructor, a constructor with <code>int</code> and <code>String</code> parameters, and getters/setters for <code>carType_no</code> and <code>car_Type</code>, along with a <code>toString()</code> method.</p>	<p>차 카테고리 테이블을 담는 VO 클래스</p>



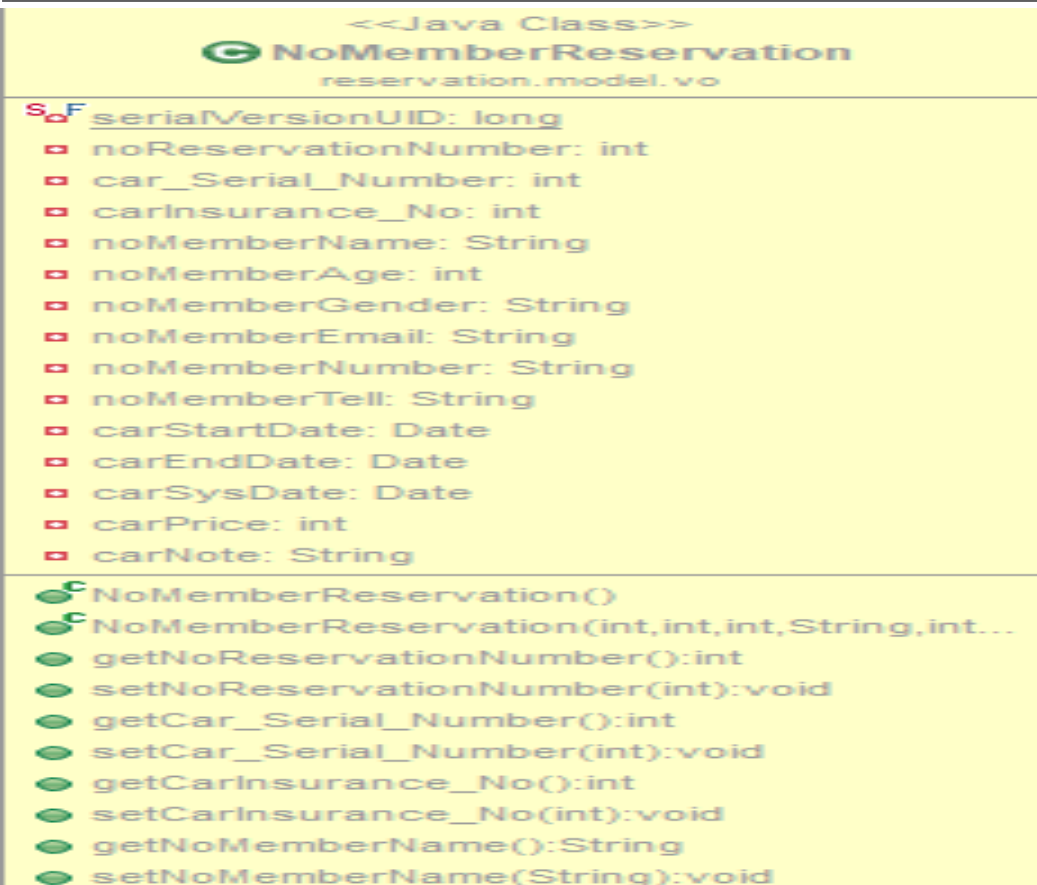
# 예약 관련 패키지 – Model -VO

클래스 다이어그램	설명
 <pre>classDiagram     class CarType {         serialVersionUID: long         car_Serial_Number: int         carType_No: int         carName: String         carNum: int         carPrice: int         carInformation: String         carCompany: String     }     CarType()     CarType(int, int, String, int, int, String, String)     getCar_Serial_Number(): int     setCar_Serial_Number(int): void     getCarType_No(): int     setCarType_No(int): void     getCarName(): String     setCarName(String): void     getCarNum(): int     setCarNum(int): void     getCarPrice(): int     setCarPrice(int): void     getCarInformation(): String     setCarInformation(String): void     getCarCompany(): String     setCarCompany(String): void     toString(): String</pre> <p>The diagram shows a Java Class named CarType in the package reservation.model.vo. It has a serialVersionUID: long and several attributes: car_Serial_Number: int, carType_No: int, carName: String, carNum: int, carPrice: int, carInformation: String, and carCompany: String. The methods include a default constructor CarType(), a parameterized constructor CarType(int, int, String, int, int, String, String), and getter/setter methods for each attribute, along with a toString() method.</p>	<p>차량 타입 테이블을 저장하는 VO 클래스</p>

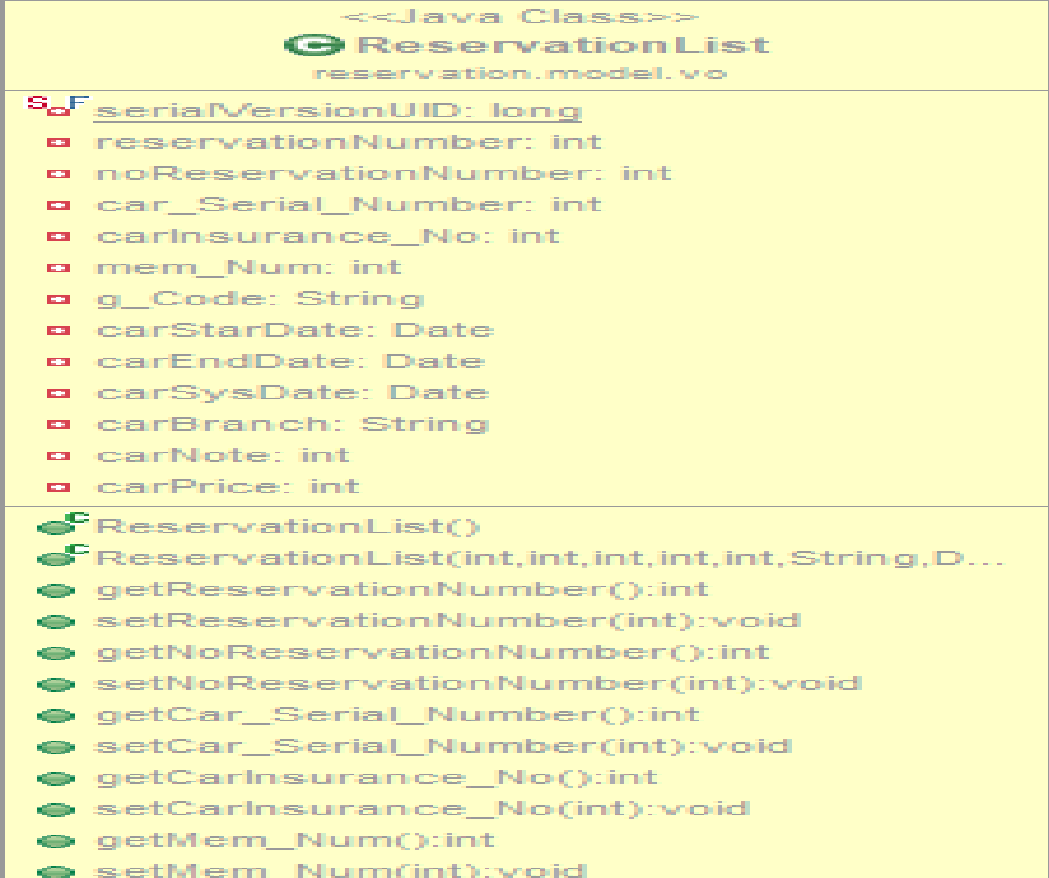
# 예약 관련 패키지 – Model -VO

클래스 다이어그램	설명
 <pre>classDiagram     class CarInsurance {         serialVersionUID: long         carInsurance_No: int         carInsurance_Name: String         carInsurance_Information: String         carInsurance_Price: int         CarInsurance()         CarInsurance(int, String, String, int)         getCarInsurance_No(): int         setCarInsurance_No(int): void         getCarInsurance_Name(): String         setCarInsurance_Name(String): void         getCarInsurance_Information(): String         setCarInsurance_Information(String): void         getCarInsurance_Price(): int         setCarInsurance_Price(int): void         getSerialVersionUID(): long         toString(): String     }</pre> <p>The diagram shows a Java class named <b>CarInsurance</b> located in the package <code>reservation.model.vo</code>. It has a <code>serialVersionUID</code> of type <code>long</code> and four attributes: <code>carInsurance_No</code> (int), <code>carInsurance_Name</code> (String), <code>carInsurance_Information</code> (String), and <code>carInsurance_Price</code> (int). The methods include a no-argument constructor, a constructor with four parameters, and getters/setters for each attribute, along with <code>toString()</code>.</p>	<p>보험 테이블을 저장하는 VO 클래스</p>

# 예약 관련 패키지 – Model -VO

클래스 다이어그램	설명
 <pre>classDiagram     class NoMemberReservation {         serialVersionUID: long         noReservationNumber: int         car_Serial_Number: int         carInsurance_No: int         noMemberName: String         noMemberAge: int         noMemberGender: String         noMemberEmail: String         noMemberNumber: String         noMemberTell: String         carStartDate: Date         carEndDate: Date         carSysDate: Date         carPrice: int         carNote: String     }     NoMemberReservation()     NoMemberReservation(int, int, int, String, int, ...)</pre> <p>The diagram shows a Java Class named <b>NoMemberReservation</b> with package <b>reservation.model.vo</b>. It contains 15 attributes: <b>serialVersionUID</b> (long), <b>noReservationNumber</b> (int), <b>car_Serial_Number</b> (int), <b>carInsurance_No</b> (int), <b>noMemberName</b> (String), <b>noMemberAge</b> (int), <b>noMemberGender</b> (String), <b>noMemberEmail</b> (String), <b>noMemberNumber</b> (String), <b>noMemberTell</b> (String), <b>carStartDate</b> (Date), <b>carEndDate</b> (Date), <b>carSysDate</b> (Date), <b>carPrice</b> (int), and <b>carNote</b> (String). The methods include a default constructor <b>NoMemberReservation()</b>, a parameterized constructor <b>NoMemberReservation(int, int, int, String, int, ...)</b>, and several getter and setter methods for the attributes.</p>	<p>비회원 예약 테이블 데이터를 저장하는 VO 클래스</p>





# 예약 관련 패키지 – Model -VO

클래스 다이어그램	설명
 <pre>classDiagram     class ReservationList {         serialVersionUID: long         reservationNumber: int         noReservationNumber: int         car_Serial_Number: int         carInsurance_No: int         mem_Num: int         g_Code: String         carStartDate: Date         carEndDate: Date         carSysDate: Date         carBranch: String         carNote: int         carPrice: int     }     ReservationList()     ReservationList(int, int, int, int, int, int, String, D...)     getReservationNumber(): int     setReservationNumber(int): void     getNoReservationNumber(): int     setNoReservationNumber(int): void     getCar_Serial_Number(): int     setCar_Serial_Number(int): void     getCarInsurance_No(): int     setCarInsurance_No(int): void     getMem_Num(): int     setMem_Num(int): void</pre> <p>UML Class Diagram for ReservationList (reservation.model.vo):</p> <ul style="list-style-type: none"><li>Attributes:<ul style="list-style-type: none"><li>serialVersionUID: long</li><li>reservationNumber: int</li><li>noReservationNumber: int</li><li>car_Serial_Number: int</li><li>carInsurance_No: int</li><li>mem_Num: int</li><li>g_Code: String</li><li>carStartDate: Date</li><li>carEndDate: Date</li><li>carSysDate: Date</li><li>carBranch: String</li><li>carNote: int</li><li>carPrice: int</li></ul></li><li>Constructors:<ul style="list-style-type: none"><li>ReservationList()</li><li>ReservationList(int, int, int, int, int, int, String, D...)</li></ul></li><li>Operations:<ul style="list-style-type: none"><li>getReservationNumber(): int</li><li>setReservationNumber(int): void</li><li>getNoReservationNumber(): int</li><li>setNoReservationNumber(int): void</li><li>getCar_Serial_Number(): int</li><li>setCar_Serial_Number(int): void</li><li>getCarInsurance_No(): int</li><li>setCarInsurance_No(int): void</li><li>getMem_Num(): int</li><li>setMem_Num(int): void</li></ul></li></ul>	<p>예약리스트 테이블 데이터를 저장하는 VO 클래스</p>


# 예약 관련 패키지 – Controller

클래스 다이어그램	설명
<pre>&lt;&lt;Java Class&gt;&gt; CarCategory reservation.controller  serialVersionUID: long  CarCategory() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</pre>	예약화면 출력 시 차량 정보 리스트를 출력해줄 수 있는 서블릿
<pre>&lt;&lt;Java Class&gt;&gt; InsuranceServlet reservation.controller  serialVersionUID: long  InsuranceServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</pre>	예약 화면 출력 시 보험 정보를 리스트에 출력해 줄 수 있는 서블릿
<pre>&lt;&lt;Java Class&gt;&gt; NoMemberReservationServlet reservation.controller  serialVersionUID: long  NoMemberReservationServlet() doGet(HttpServletRequest, HttpServletResponse): void doPost(HttpServletRequest, HttpServletResponse): void</pre>	VO 에 담겨있는 데이터를 DB에 보내주는 서블릿

# 예약 관련 패키지 – Model -Service

클래스 다이어그램	설명
<p>&lt;&lt;Java Class&gt;&gt;</p> <p> ReservationService</p> <p>reservation.model.service</p>	
<p> ReservationService()</p> <p> insertNoMemberReservation(NoMemberReservation, C...</p> <p> carTypeList(int):ArrayList&lt;CarType&gt;</p> <p> insuranceList():ArrayList&lt;CarInsurance&gt;</p>	차량 예약에 관한 것을 담당하는 Service

# 예약 관련 패키지 – Model -DAO

클래스 다이어그램	설명
 <pre>classDiagram     class ReservationDao {         &lt;&lt;Java Class&gt;&gt;         reservation.model.dao         +prop: Properties         +ReservationDao()         +insertNoMeberReservation(Connection, NoMemberRe...)         +carTypeList(Connection, int): ArrayList&lt;CarType&gt;         +insuranceList(Connection): ArrayList&lt;CarInsurance&gt;     }</pre> <p>The diagram shows a class named <b>ReservationDao</b> in the package <code>reservation.model.dao</code>. It has a property <code>prop: Properties</code> and four methods: <code>ReservationDao()</code>, <code>insertNoMeberReservation(Connection, NoMemberRe...)</code>, <code>carTypeList(Connection, int): ArrayList&lt;CarType&gt;</code>, and <code>insuranceList(Connection): ArrayList&lt;CarInsurance&gt;</code>.</p>	<p>차량 예약을 담당하는 DAO</p>

# 예약 관련 패키지 – View

---

## InsuranceReservation.jsp

- 차량 보험 관련 된 정보를 볼 수 있는 뷰

## LongReservation.jsp

- 장기 예약을 진행 할 수 있는 뷰

## ShortReservation.jsp

- 단기 예약을 진행 할 수 있는 뷰



# 예약 관련 패키지 – View

---

## NoMemberReservation.jsp

- 비회원 예약 시 사용하는 뷰

## CarInformation.jsp

- 각 차에 대한 정보를 볼 수 있는 뷰

## Reservation.jsp

- 회원 자신의 예약 내역을 볼 수 있는 뷰

---

# Thank You