[0/188] BaseFilter.java

这个文件是一个抽象类，命名为 BaseFilter。它实现了 Filter 接口。

该类包含以下方法：

- destroy()：空方法，表示销毁过滤器。
- doFilter(ServletRequest req, ServletResponse resp, FilterChain chain)：将 ServletRequest 和 ServletResponse 转换为 HttpServletRequest 和 HttpServletResponse 对象，并调用抽象方法 doFilter(HttpServletRequest req, HttpServletResponse resp, FilterChain chain) 进行具体的过滤操作。
- init(FilterConfig config)：空方法，表示初始化过滤器。
- protected abstract void doFilter(HttpServletRequest req, HttpServletResponse resp, FilterChain chain)：抽象方法，用于实现具体的过滤操作。

此文件的代码主要用于提供一个基础的过滤器类，其他具体的过滤器类可以继承该类并实现抽象方法 doFilter()，以实现自定义的过滤功能。

[1/188] BaseServlet.java

这是一个名为 BaseServlet 的抽象类，它继承自 HttpServlet 类。它包含了一些常量和方法：

- 常量：MSG_KEY 和 ERROR_KEY，用于设置 session 中的消息和错误的键名。
- 方法：
    - doGet：重写了 HttpServlet 的 doGet 方法，调用了 doAction 方法。
    - doPost：重写了 HttpServlet 的 doPost 方法，调用了 doAction 方法。
    - doAction：一个空实现的方法，用于处理具体的业务逻辑，在子类中需要实现。
    - addMessage：向 session 中添加消息。
    - addError：向 session 中添加错误。
    - getCurrentUser：获取当前用户，从 session 中获取。

该类是一个可以被继承的基类，用于实现具体的 servlet 类。它提供了一些通用的方法和常量，可以被子类继承和调用。

[2/188] Constant.java

这个程序文件是一个名为 Constant 的 Java 类。它包含了一些常量和一个静态方法 join。其中，常量包括 CURRENT_USER、CURRENT_QUESTION、CURRENT_TESTPERSONNEL 和 CURRENT_EXAM，分别表示当前用户、当前问题、当前测试人员和当前考试。静态方法 join 接受一个字符串数组作为参数，并将数组中的字符串以逗号分隔的形式连接起来。如果数组为空，方法返回 null。

[3/188] Db.java

这是一个名为"Db.java"的代码文件，位于""目录下。该文件定义了一个名为 Db 的类，用于数据库的连接和操作。

该类包含以下内容：

- 一个静态代码块，在类加载时通过反射实例化一个 com.mysql.cj.jdbc.Driver 对象。
- 一个静态方法 getConnection()，用于获取数据库连接，并设置自动提交事务。
- 一个静态方法 getGeneratedInt()，接收一个 Statement 对象作为参数，并返回最后插入的自增 ID。

数据库连接的 URL、用户名和密码等信息被硬编码在类的静态常量中，用于连接名为"mbti"的 MySQL 数据库。数据库连接使用的是 MySQL 的 JDBC 驱动。

该类的作用是提供数据库连接和一些基本的操作方法，供其他类使用。

[4/188] EncodeFilter.java

这个程序文件是一个过滤器类，名称为 EncodeFilter。它是一个 Java 类，属于 com.qst 包。该过滤器类继承自 BaseFilter 类，并使用@WebFilter 注解标记，表示它是一个 Web 过滤器。该过滤器的功能是设置请求和响应的字符编码为 utf-8，并将请求和响应交给 FilterChain 继续处理。它的过滤范围是以".jsp"、".action"和".do"结尾的 URL。

[5/188] EntityToMapConverter.java

这个程序文件是一个名为`EntityToMapConverter.java`的 Java 类文件。它包含了一个静态工具类`EntityToMapConverter`，其中提供了两个静态方法。

1. `convertEntityToMap`方法接受一个实体对象作为参数，并将该对象转换成一个 Map 对象。它通过使用反射获取实体类的字段，并将字段名和字段值作为键值对存储在 Map 中。如果实体对象是`BaseEntity`的实例，还会将`id`字段和其对应的值存储到 Map 中。
2. `convertEntityListToMapList`方法接受一个实体对象的 List 作为参数，并将其中每个对象转换成一个 Map 对象，并将所有的 Map 对象存储在一个 List 中。该方法调用了上面提到的`convertEntityToMap`方法来实现转换。

[6/188] ExamException.java

这是一个 Java 文件，名称为 ExamException.java。它位于 com.qst 包下的 mbti 项目中。该文件定义了一个自定义异常类 ExamException，该类继承自 RuntimeException。该类包含了三个构造函数，分别用于创建无参数异常、带有错误信息的异常和带有错误信息和原因的异常。

[7/188] MysqlCjAbandonedConnectionCleanupListener.java

这个文件是一个 Java 类文件，命名为 MysqlCjAbandonedConnectionCleanupListener.java。它是一个实现了 ServletContextListener 接口的监听器类。它用于在 ServletContext 初始化和销毁时进行一些操作。在初始化阶段，它会在控制台打印一条信息。在销毁阶段，它会打印一条信息并释放 JDBC 连接资源。具体地，它会遍历所有的 JDBC 驱动程序，并将它们从 DriverManager 中注销。然后，它会调用 AbandonedConnectionCleanupThread 类的 uncheckedShutdown 方法，用于关闭未关闭的连接。如果在销毁过程中发生异常，它会打印异常信息并报告销毁工作异常。

[8/188] package-info.java

这个程序文件是位于 mbti.zip.extract/mbti/src/main/java/com/qst 路径下的 package-info.java 文件。它的主要作用是定义了一个 com.qst 的包。该包可能包含了与 MBTI（Myers-Briggs Type Indicator）相关的类和功能。

[9/188] RequestUtil.java

这个程序文件是一个名为 RequestUtil 的 Java 类，位于 com.qst 包下。它提供了一些静态方法，用于获取 HttpServletRequest 对象中的参数值并转换为不同类型（String、int、int 数

组、double、Date、Timestamp）。此外，还提供了一个私有的 isBlank 方法，用于判断字符串是否为空或只包含空格。

[10/188] SecurityFilter.java

这是一个名为"SecurityFilter.java"的 Java 程序文件。它是一个 Servlet 过滤器，用于过滤对指定 URL 模式的 HTTP 请求，并进行安全性验证。该过滤器基于基类"BaseFilter"，实现了"Filter"接口。

该过滤器通过使用"Map"来定义允许访问的路径和相应的用户类型。在过滤器中，它使用请求的 URI 来判断所请求的路径，并与当前用户的类型进行比较。如果当前用户不是管理员，并且请求的路径没有与当前用户的类型匹配，将重定向到"/deny.jsp"页面。否则，请求将继续传递到下一个过滤器或目标资源。

该文件还包括一些导入语句和构造函数，用于初始化路径映射。它还重写了"doFilter"方法，这是 Servlet 过滤器中执行实际过滤逻辑的方法。

总而言之，"SecurityFilter.java"是一个用于进行安全性过滤的 Servlet 过滤器，它根据用户类型和请求路径来决定是否允许访问。

[11/188] SMMSUploader.java

这个程序文件是一个名为 SMMSUploader 的 Java 类文件。它包含一个名为 upload 的方法，该方法接收一个文件作为参数，并返回一个字符串。这个方法的作用是将文件上传到 SM.MS 图片托管服务，并返回上传后的图片 URL。该类中还定义了一些常量，如上传 URL 和 API 密钥。

[12/188] Test.java

这是一个名为 Test.java 的 Java 程序文件，位于 目录下。该程序打印出一个字符串变量 uri 中斜杠"/"的索引位置，并将其输出到控制台。

[13/188] WebUtil.java

这个程序文件是一个名为 WebUtil 的 Java 类，位于 com.qst 包下。它包含了一些静态方法用于在 Web 应用程序中进行请求转发和重定向操作。

其中的 forward 方法接收 HttpServletRequest、HttpServletResponse 和一个 URL 参数，用于执行请求转发操作。该方法会对 URL 进行编码处理，并将请求和响应对象传递给

getRequestDispatcher 方法，最后执行 forward 方法实现请求转发。

redirect 方法有两个重载版本。一个重载版本接收 HttpServletRequest、HttpServletResponse 和一个 URL 参数，另一个重载版本只接收 HttpServletResponse 和一个 URL 参数。这两个方法都用于执行重定向操作。

在重定向方法中，如果传入的 URL 以斜杠“/”开头，则会使用 getServletContext().getContextPath()获取项目的根路径，并与 URL 拼接。然后，对 URL 进行编码处理，并使用 sendRedirect 方法执行重定向操作。

注释中提到，根路径的确定方式取决于服务器或浏览器的处理方式。

[14/188] LoginServlet.java

这个文件是一个 Java Servlet 类，名为 LoginServlet.java。它位于 目录下。该类处理用户登录请求，通过接收用户提交的用户名和密码，调用 UserService.login 方法进行验证。如果验证成功，将用户信息设置到会话中，然后根据用户类型判断是否为测试人员，并将测试人员信息设置到会话中。最后，通过重定向将用户导航到主页。如果发生了 ExamException 异常，则将异常信息设置到请求属性，并进行页面转发到登录页面。

[15/188] LogoutServlet.java

这个程序文件是一个名为 LogoutServlet.java 的 Java 类，它位于 com.qst.action 包中。该类继承自 BaseServlet 类，并且带有一个@WebServlet 注解。它覆盖了 BaseServlet 类的 doAction 方法，并在其中实现了用户登出的功能。具体操作是通过调用 req.getSession().invalidate()方法使当前会话失效，然后使用 WebUtil 类调用 redirect 方法将用户重定向到/login.jsp 页面。该类的作用是处理用户登出的请求。

[16/188] PasswordServlet.java

这个文件是一个 Java 源代码文件，其文件路径是 PasswordServlet.java。该文件是一个 Servlet 类，继承自 BaseServlet 类。它处理密码相关的操作，包括修改密码和展示修改密码页面。

在 doGet 方法中，该 Servlet 将请求转发到/password.jsp，以展示修改密码页面。

在 doPost 方法中，该 Servlet 获取当前用户信息，并获取旧密码、新密码和确认密码。如果确认密码与新密码不一致，则将错误消息添加到请求中，并将请求转发到/password.jsp。否则，

该 Servlet 调用 userService 对象的 changePassword 方法来修改密码，并将成功消息添加到请求中，最终将请求重定向到/passwordmsg.jsp 页面。

该文件还导入了一些其他的类，如 BaseServlet、Constant、ExamException、RequestUtil、WebUtil、User、IUserService 和 ServiceFactory。

[17/188] AIQuiz/BaiduAIAPI.java

这是一个名为 BaiduAIAPI.java 的文件，包含了一个名为 BaiduAIAPI 的 Java 类。该类是用来调用百度 AI 接口生成文字的 API。该类中有一个公共静态方法 exec，它接受一个名为 allAnswers 的参数，并返回一个字符串作为生成的文字结果。在方法内部，它使用了 Java 的网络连接 API 来发送 HTTP 请求，通过百度 AI 接口来获取访问令牌，并执行聊天请求，最终解析聊天结果并返回。该类还包含了一些常量，如百度 AI 密钥和访问令牌等。

[18/188] AIQuiz/ChatServlet.java

这个程序文件是一个 Java servlet，它用于处理聊天的 POST 请求和 GET 请求。在 POST 请求中，它从请求参数中获取 10 个问题的答案，并将答案传递给 BaiduAIAPI 进行处理。然后，它使用 TextRankKeyword 来提取关键词，并将关键词传递给 WordToImageConverter 进行转换成图像的二进制数据。最后，它将图像的 Base64 编码和聊天结果设置为请求属性，并将请求转发到 chatResult.jsp 进行显示。在 GET 请求中，它直接将请求转发到 chatResult.jsp。此外，它还提供了一个静态方法 saveBase64ImageToFile，用于将 Base64 编码的图像数据保存到文件中。

[19/188] AIQuiz/TextRankKeyword.java

这个程序文件名为 TextRankKeyword.java。它是一个关键词提取工具类，使用 TextRank 算法来提取关键词。它通过将标题和内容进行分词，并根据词语之间的关联性计算关键词的重要程度。提取的关键词数量为 10 个。该类还包含一个 main 方法，用于演示如何使用这个关键词提取工具类。

[20/188] AIQuiz/WordToImageConverter.java

该文件是一个 Java 类文件，名为 WordToImageConverter。该类位于包 com.qst.action.AIQuiz 下。

这个类实现了文字转图像的 API。它具有以下主要方法：

- getAccessToken(): 获取访问令牌的方法。
- getTextToImage(String accessToken, String textInput): 将文本转换为图像的方法。
- createConnection(String urlString, String method): 创建网络连接的方法。
- word2pix(String text): 程序的入口方法，将输入的文本转换为图像。

该类还包含了一些常量变量，包括 AK 和 SK，分别表示百度 AI 的 Access Key 和 Secret Key。

该文件使用了 Java 的标准库和第三方库，例如 java.awt.image、java.io、java.net、java.nio.file、java.util.Base64、javax.imageio 等。这些库负责处理图像数据、网络连接和文件操作。

总之，该文件是一个实现文字转图像功能的 Java 类。它封装了百度 AI 的 API，通过访问令牌和文本输入，将文本转换为图像数据。

[21/188] AssessmentType/CreateServlet.java

这个文件是一个 Java 源代码文件，文件路径是`AssessmentType/CreateServlet.java`。它所属的包是`com.qst.action.AssessmentType`。这个文件是一个 servlet，继承自`BaseServlet`类，通过注解`@WebServlet("/assessment/create.action")`映射了一个 URL 路径`/assessment/create.action`。在`doAction`方法中，它调用了`WebUtil.forward`方法来将请求转发到`/assessment/create.jsp`页面。

[22/188] AssessmentType/DeleteServlet.java

该文件是一个用于处理删除考核类型的 Servlet。它继承自 BaseServlet，实现了 doAction 方法，该方法会在接收到请求时被调用。在 doAction 方法中，它首先通过 RequestUtil 工具类获取请求中的考核类型 ID，并将其传递给 assessmentService 对象的 deleteAssessment 方法进行删除操作。如果删除成功，它会向请求中添加一条成功消息；如果删除失败，它会向请求中添加一条错误消息。最后，它会通过 WebUtil 工具类进行页面重定向，将用户导航到考核类型列表页面。

[23/188] AssessmentType/EditServlet.java

这是一个 Java 源代码文件，文件名为 EditServlet.java。它是一个用于编辑测评类型的 Servlet，在接收前端发送的请求后，通过调用 service 层的方法查询数据，并将查询结果返回

给前端页面。它继承自 BaseServlet 类，并且使用@WebServlet 注解将 URL 映射到"/assessment/edit.action"。在 doAction 方法中，它首先从请求参数中获取 id 值，然后调用 assessmentService 的 findAssessmentById 方法查询对应 id 的测评类型数据，将查询结果设置为请求属性"assessment"，最后通过 WebUtil.forward 方法将请求转发到"/assessment/edit.jsp"页面。

[24/188] AssessmentType/ListServlet.java

这个程序文件是一个 Java 类文件，位于`com.qst.action.AssessmentType`包下。它是一个 Servlet 类，扩展了`BaseServlet`类，并在`/assessment/list.action`路径下处理 HTTP 请求。

该类包含了一个私有的`IAssessmentService`类型的成员变量`assessmentService`，该成员变量通过`ServiceFactory.getService(IAssessmentService.class)`获取。在`doAction`方法中，调用`assessmentService`的`findAllAssessment`方法，获取考核类型列表，并将列表设置为请求属性`assessmentList`。最后，调用`WebUtil.forward`方法将请求转发到`/assessment/list.jsp`页面。

[25/188] AssessmentType/SaveServlet.java

这个程序文件是一个用于保存考核类型信息的 Servlet。它包含一个名为 SaveServlet 的类，继承自 BaseServlet。在 doAction 方法中，它接收来自前端的请求，通过 HttpServletRequest 获取请求参数，并通过调用 IAssessmentService 的 saveAssessment 方法将考核类型信息保存到数据库中。如果保存成功，会向 HttpServletRequest 添加一条成功的消息，并通过 WebUtil 的 redirect 方法跳转到显示详情的请求。如果保存失败，会将错误信息添加到 HttpServletRequest，然后通过 WebUtil 的 forward 方法跳转回创建页面，并将考核类型信息回传给前端。

[26/188] AssessmentType/UpdateServlet.java

这个程序文件是一个 Java 类文件，位于 AssessmentType/UpdateServlet.java 路径下。它是一个用于更新考核类型的 Servlet 类。该类继承自 BaseServlet 并通过@WebServlet 注解映射到"/assessment/update.action"路径。在 doAction 方法中，它接收并处理 HttpServletRequest 和 HttpServletResponse 参数。它从请求中获取 id、title、cost 和 status 参数，并创建一个 AssessmentType 对象。然后，它调用 assessmentService 的 updateAssessment 方法更新考核类型信息，并根据操作结果进行相应的页面跳转或错误处理。

[27/188] AssessmentType/ViewServlet.java

这段代码是一个 Java Servlet，名为 ViewServlet，它处理来自`/assessment/view.action`网址的请求。它通过调用`AssessmentServiceImpl`的`findAssessmentById`方法来查找特定考核类型的信息，并将该信息保存在 request 对象的属性中，然后将请求转发到`/assessment/view.jsp`页面进行显示。

[28/188] exam/BeginServlet.java

该代码文件是 Java 源代码文件，位于`exam/`目录下。代码使用了`com.qst`包中的类和常量。

这是一个名为`BeginServlet`的类，它扩展了`BaseServlet`类。类中包含了私有成员变量`assessmentService`、`examService`和`scheduleService`，分别是`IAssessmentService`、`IExamService`和`IScheduleService`接口的实例。

在`doAction`方法中，通过访问`req`（HttpServletRequest 对象）中的参数来获取`testPersonnel`（TestPersonnel 对象）和`scheduleId`（整数值）。然后调用`examService.begin(testPersonnel, scheduleId)`方法生成一个`exam`（Exam 对象），并将其存储在`req`的会话中。最后，通过调用`WebUtil.redirect(req, resp, "/exam/exam.action")`方法将请求重定向到`/exam/exam.action`路径。

如果在生成试题过程中发生`ExamException`异常，会打印异常堆栈信息并通过调用`addError(req, ex.getMessage())`方法将错误信息添加到`req`中。然后通过调用`WebUtil.redirect(req, resp, "/exam/list.action")`方法将请求重定向到`/exam/list.action`路径。

此代码的主要功能是通过访问“/exam/begin.action”路径，根据给定的`testPersonnel`和`scheduleId`生成一个试题，并将其存储在会话中，然后将请求重定向到生成的试题页面。如果在生成试题过程中发生异常，会将错误信息添加到请求中，并将请求重定向到试题列表页面。

[29/188] exam/EndServlet.java

该文件是一个 Java 源代码文件，位于 exam/EndServlet.java。它是一个处理结束考试请求的 servlet 类。

该 servlet 类继承自 BaseServlet，覆盖了父类的 doAction 方法，实现了

HttpServletRequest 和 HttpServletResponse 对象的处理逻辑。

在 doAction 方法内，它首先从 HttpServletRequest 的 session 中获取当前考试的 Exam 对象，然后通过调用 examService 的 end 方法结束考试。接着根据考试的结果生成一个结果页面的路径，并将路径和考试的得分保存到 HttpServletRequest 的 attribute 中。

如果在结束考试的过程中发生了 ExamException 异常，它会输出异常的堆栈轨迹，并将错误信息保存到 HttpServletRequest 的 attribute 中。

最后，它通过 WebUtil 类的 forward 方法将 HttpServletRequest 和 HttpServletResponse 对象转发到/exam/end.jsp 页面。

该 servlet 类还引入了一些其他类和常量，如 com.qst.Constant 中的一些静态常量以及 com.qst.entity、com.qst.service 和 com.qst.WebUtil 等其他类。

[30/188] exam/ExamServlet.java

这个程序文件是一个 Java 源文件，路径为`exam/ExamServlet.java`。它定义了一个名为`ExamServlet`的 Servlet 类，继承自`BaseServlet`类，并且被`@WebServlet("/exam/exam.action")`注解标记为一个 Servlet。这个类包含了一个`doAction`方法，用于处理 HTTP 请求。在这个方法中，它做了以下几件事情：

1. 获取当前考试的信息(`Exam`)，从会话属性(`req.getSession().getAttribute(CURRENT_EXAM)`)中获取。
2. 获取前端返回的答案(`answer`)，通过`RequestUtil.getIntArray`方法处理。
3. 设置当前问题的答案(`eq.setAnswer(answer)`)。
4. 判断传入的`index`参数的值，根据不同的情况进行相应的操作：
   a. 如果`index`为-3，则转发到"/exam/end.action"页面。
   b. 如果`index`为-2，则将题目序号加一。
   c. 如果`index`为-1，则将题目序号减一。
   d. 如果`index`为其他值，则将题目序号设置为指定的值。
5. 获取当前问题的信息(`eq = exam.getQuestion()`)。
6. 检查题目的选项是否存在，如果不存在，则从数据库中加载选项，并打乱选项的顺序。
7. 根据答案的记录，确定哪些选项需要被标记为选中状态。
8. 将请求转发到"/exam/exam.jsp"页面。

这个 Servlet 类的作用是处理考试相关的操作，包括获取题目、提交答案等。

[31/188] exam/ListServlet.java

这是一个名为`ListServlet.java`的 Java 源代码文件，位于`exam/`目录下。

该文件定义了一个名为`ListServlet`的 Java 类，该类继承了`BaseServlet`类，并使用了`@WebServlet`注解将该 Servlet 映射到路径`/exam/list.action`。

该类包含了三个成员变量`assessmentService`、`examService`和`scheduleService`，它们是`IAssessmentService`、`IExamService`和`IScheduleService`接口的实例，并通过`IAssessmentService.class`、`IExamService.class`和`IScheduleService.class`获取实例对象。

该类重写了父类的`doAction`方法，该方法会根据当前用户在会话中的属性`Constant.CURRENT_TESTPERSONNEL`获取到`TestPersonnel`对象，并调用`examService`的`findScheduleByTestPersonnel`方法获取到该用户的考试安排列表，并将其设置到 Request 的属性`scheduleList`中。

最后，通过调用`WebUtil`的`forward`方法将请求转发到"/exam/list.jsp"页面。

[32/188] exam/ResultServlet.java

这是一个名为 ResultServlet.java 的 Java 源代码文件。它位于 exam/目录下。代码使用了 Servlet 注解@WebServlet("/exam/result.action")，表示这是一个处理/exam/result.action 请求的 Servlet。

该 Servlet 类继承了 BaseServlet 类，并重写了 doAction 方法。doAction 方法从请求参数中获取 id，然后调用 examService 的 examResult 方法获取对应 id 的考试结果。接下来将考试结果转发到/exam/result.jsp 页面进行展示。

代码中还使用了 Constant.java、BaseServlet.java、ExamException.java、RequestUtil.java、WebUtil.java 等其他类，并且通过 ServiceFactory 获取了 IAssessmentService 类、IExamService 类和 IScheduleService 类的实例。这些类和实例的具体功能需要进一步查看源代码才能确定。

[33/188] movie/AddServlet.java

该文件是一个 Java 类文件，包名为 com.qst.action.movie，类名为 AddServlet。这个类继承

自 BaseServlet，并且被@WebServlet("/movie/add.action")注解标记为一个 Servlet，允许支持文件上传。

在 doAction 方法中，首先创建了一个 Movie 对象，并根据请求参数设置 Movie 对象的属性。之后，通过 request.getPart 方法获取到上传的文件，并利用输入流将文件内容写入到本地文件。然后，使用 SMMSUploader 类将本地文件上传到服务器，并获取到上传后的文件 URL。最后，将 Movie 对象保存到数据库中，并通过 WebUtil 工具类进行页面跳转。

在异常捕获部分，如果出现 ExamException 异常，则将异常信息设置到请求属性中，并通过 WebUtil.forward 方法将请求转发到"/movie/manage.jsp"页面进行错误展示。

整体来说，这个类的作用是处理电影信息的添加操作，包括获取前端传递的电影信息、上传电影海报、保存电影信息到数据库，并提供错误处理和页面跳转功能。

[34/188] movie/DeleteServlet.java

这是一个名为 DeleteServlet.java 的 Java 文件，它位于 movie 目录下。该文件是一个 Servlet 类，用于处理电影删除操作。它继承自 BaseServlet 类，实现了 doAction 方法。在 doAction 方法中，它接收到 HTTP 请求后，从请求参数中获取电影 ID，然后调用 movieService 的 deleteMovie 方法来删除对应的电影。最后，它通过 WebUtil 类将请求重定向到/movie/list.action 的 URL 上。

[35/188] movie/EditServlet.java

这是一个名为 EditServlet.java 的文件，位于 movie 路径下。

该文件是一个 Java Servlet 文件，其作用是处理与电影编辑相关的 HTTP 请求。它包含一个 doAction 方法，该方法接收 HttpServletRequest 和 HttpServletResponse 对象作为参数，用于处理具体的请求逻辑。

在 doAction 方法中，文件首先调用 RequestUtil 工具类的 getInt 方法来获取请求中的电影 ID，并通过 movieService 获取对应 ID 的电影信息。然后，从请求参数中获取名称为 date 的字符串，并将其转换为 Timestamp 类型的时间戳。

最后，文件将获取到的电影对象设置到 request 的属性中，并使用 WebUtil 工具类的 forward 方法将请求转发到/movie/edit.jsp 页面进行渲染。

该文件的作用是实现了一个用于编辑电影页面的 Servlet，并提供了获取电影信息和时间戳转换

的功能。

[36/188] movie/ListServlet.java

这个文件名为 ListServlet.java 的文件是一个 Java servlet 文件。它是一个电影列表页面的后端处理逻辑。在该文件中，有一个名为 ListServlet 的类，它是 BaseServlet 类的子类。该类处理来自前端的请求并生成适当的响应。

在 doAction 方法中，封装了获取电影信息、检查用户是否预订电影、设置相应参数等操作。它首先通过 IMovieService 接口获取所有电影的列表，然后将电影列表转换为 Map 类型的列表。接下来，它通过比较用户的类型来决定将用户重定向到管理页面还是电影列表页面。

该文件使用一些其他的类和接口，例如 BaseServlet，Constant，EntityToMapConverter，WebUtil 等。这些类和接口提供了与电影和用户相关的服务和功能，如获取电影列表、将对象转换为 Map，处理用户请求等。

[37/188] PersonalityDimension/DeleteServlet.java

这是一个名为"DeleteServlet.java"的 Java 源代码文件。它位于"PersonalityDimension"目录下。

这个文件实现了一个 servlet，它处理对"/dimension/delete.action"的 HTTP 请求。在处理请求时，它首先从请求中获取一个名为"id"的整数参数，然后使用 IDimensionService 接口的实现类调用 deleteDimension 方法来删除相应的问卷维度信息。如果删除成功，则向请求中添加一条消息，表示问卷维度已删除。如果删除失败，则向请求中添加一个错误消息。最后，它重定向到"/dimension/list.action"页面，并附带一个名为"assessmentId"的整数参数。

总之，这个文件实现了一个用于删除问卷维度信息的 Servlet，并提供了一些与请求相关的功能。

[38/188] PersonalityDimension/EditServlet.java

该文件是一个名为"EditServlet.java"的 Java Servlet 文件。它位于"PersonalityDimension"目录下。

这个 Servlet 类是一个继承自"BaseServlet"的类，并且使用了@WebServlet 注解，将 URL 路径"/dimension/edit.action"映射到该 Servlet 中。

在 doAction 方法中，它首先通过调用 RequestUtil.getInt(req, "id")获取请求参数中名为"id"的整数值，然后通过 dimensionService 对象调用 findDimensionById 方法来根据问卷维度 ID 查找问卷维度信息，将结果设置到 HttpServletRequest 的 attribute 中，然后通过 WebUtil.forward 方法将请求转发到"/dimension/edit.jsp"页面进行显示。

[39/188] PersonalityDimension/ListServlet.java

这个文件是一个 Java Servlet 文件，用于处理关于人格维度的操作。它包含了两个方法：doCreate 和 doList。doCreate 方法用于处理添加问卷维度的跳转控制，通过查找考核类型信息并转发到 create.jsp 页面。doList 方法用于查询考核类型列表和相应的问卷维度列表，并转发到 list.jsp 页面。该文件还包含了两个成员变量：assessmentService 和 dimensionService，它们分别是 IAssessmentService 和 IDimensionService 的实现类的实例。整个文件通过@WebServlet 注解映射到"/dimension/list.action"路径。

[40/188] PersonalityDimension/SaveServlet.java

这个程序文件是一个保存问卷维度信息的 Servlet。它接收请求参数，创建一个 PersonalityDimension 对象，并使用 DimensionService 将该对象保存到数据库中。如果保存成功，它会添加一个提示信息，然后重定向到显示问卷维度详情的请求。如果保存失败，它会将错误信息添加到请求属性中，并转发到创建问卷维度的页面。

[41/188] PersonalityDimension/UpdateServlet.java

这是一个名为 UpdateServlet.java 的文件，位于 PersonalityDimension 目录下。

代码的主要功能是处理 HTTP 请求并更新数据库中的问卷维度信息。它使用了 BaseServlet 作为父类，并在 doAction 方法中实现了具体的逻辑。在方法中，它根据请求参数创建一个 PersonalityDimension 对象，并调用 dimensionService.updateDimension 方法更新维度信息，在更新成功后向请求中添加消息，并重定向到另一个 URL 以显示修改后的维度信息。如果在更新过程中发生异常，它会向请求中添加错误信息，并将维度对象和相关的评估对象设置为请求属性，最后转发到另一个 JSP 页面以显示错误信息和允许用户重新编辑维度信息。

[42/188] PersonalityDimension/ViewServlet.java

这是一个 Java 程序文件，文件名为 ViewServlet.java。该程序文件属于 com.qst.action.PersonalityDimension 包。

该程序文件是一个 Servlet，通过继承 BaseServlet 类来处理 HTTP 请求。它使用 @WebServlet 注解来指定 URL 路径为"/dimension/view.action"。

doAction 方法是 Servlet 的主要逻辑方法，它接受 HttpServletRequest 和 HttpServletResponse 对象作为参数，并抛出 ServletException 和 IOException。

在 doAction 方法中，通过 dimensionService 对象调用 findDimensionById 方法，根据请求参数中的"id"来查找对应的 PersonalityDimension 对象。然后将查找到的 PersonalityDimension 对象作为属性设置到请求对象 req 中。

最后，使用 WebUtil 类的 forward 方法将请求转发到"/dimension/view.jsp"页面进行显示。

[43/188] question/DeleteServlet.java

该程序文件是一个 Servlet 类，位于`question/DeleteServlet.java`文件中。该类负责处理删除题目的请求。它继承自`BaseServlet`，并使用`@WebServlet("/question/delete.action")`注解声明其对应的 URL 路径为"/question/delete.action"。

该类包含一个私有成员变量`questionService`，它是一个`IQuestionService`接口的实例，通过`ServiceFactory.getService(IQuestionService.class)`方法获取。

该类重写了`doAction(HttpServletRequest req, HttpServletResponse resp)`方法，用于处理 HTTP 请求。在方法中，它首先通过`RequestUtil.getInt(req, "id")`方法获取请求参数"id"的值，并根据该 id 调用`questionService.findById(id)`方法获取对应的题目实例。

之后，它尝试调用`questionService.delete(id)`方法进行题目的删除操作，并在成功删除后向请求中添加一条消息"题目已删除"。如果在删除过程中捕获到`ExamException`异常，则向请求中添加一条错误信息。

最后，它根据题目的属性组装 URL，并使用`WebUtil.redirect(req, resp, url)`方法进行重定向，将用户导航到题目列表的页面。

[44/188] question/DimensionServlet.java

这是一个名为 DimensionServlet.java 的 Java 程序文件。它是一个 Servlet 类，用于处理与维度相关的操作。该类继承自 BaseServlet，并且通过@WebServlet 注解映射到"/question/dimension.action"路径。

该类的主要功能包括：

- 通过调用数据库中的数据，将关联的问卷维度保存到数据库。
- 根据传递的参数判断是进行保存操作还是编辑操作，并调用相应的方法。
- 编辑方法 edit 获取题目 ID，并根据题目 ID 查询题目内容和相关的问卷维度，将其存储到 request 对象中，然后跳转到 dimension.jsp 页面进行展示。
- 保存方法 save 获取题目 ID 和相应的问卷维度 ID，将其关联并保存到数据库，然后重定向到 question/view.action 页面进行展示。

该类还依赖于其他类和接口，如 BaseServlet、RequestUtil、WebUtil、PersonalityDimension、Question，以及 IDimensionService 和 IQuestionService 接口。它通过 ServiceFactory 类获取相应的服务实例。

总体而言，该类用于处理与问卷维度相关的操作，包括保存和编辑。

[45/188] question/EditServlet.java

这是一个名为"EditServlet.java"的 Java 程序文件。该文件位于"question"目录下。该文件的功能是用于处理用户点击修改按钮时的操作。它是一个 Servlet 类，继承自 BaseServlet 类。类中包含两个成员变量 assessmentService 和 questionService，分别是 IAssessmentService 和 IQuestionService 的实例。该类实现了 doAction 方法，用于执行具体的修改操作。在方法中，通过调用 questionService 的 findById 方法获取问题的详细信息，通过调用 questionService 的 findChoices 方法获取问题的选项信息。然后，通过 assessmentService 的 findAssessmentById 方法获取问题所属评估类型的详细信息。最后，将这些信息设置到 HttpServletRequest 对象的属性中，并通过 WebUtil 类的 forward 方法将请求转发到"/question/edit.jsp"页面。

[46/188] question/ListServlet.java

这个程序文件是一个 Java servlet 文件，位于`com.qst.action.question`包下，文件名为`ListServlet.java`。该 servlet 处理来自前端的请求，根据不同的请求执行不同的操作。

具体而言，该 servlet 实现了`doAction`方法，该方法根据请求中的参数判断用户是否点击了添加按钮。如果点击了添加按钮，则调用`doCreate`方法进行添加操作；否则，调用`doList`方法进行查询操作。

在`doList`方法中，该 servlet 会获取所有的考核类型，并将其封装到`assessmentList`中，然后

根据请求中的参数创建`QuestionQueryParam`对象，并调用`questionService`的`find`方法获取相应的问题列表，并将其封装到 request 对象中。最后，通过请求转发将请求转发到`question/list.jsp`页面。

在`doCreate`方法中，该 servlet 首先创建一个`QuestionQueryParam`对象，然后通过考核类型 ID 查找考核的信息，并将其存放到 request 对象中，用于页面中内容的显示。接下来，创建一个问题对象并设置相关属性，然后通过请求转发将请求转发到`question/create.jsp`页面。

[47/188] question/QuestionHelper.java

该程序文件是一个帮助类，位于 com.qst.action.question 包下。它提供了两个静态方法用于在 Servlet 层获取前端传来的数据。

createQuestion 方法接收 HttpServletRequest 对象作为参数，从请求参数中获取题目的各个属性，并将其设置给一个 Question 对象，最后返回该 Question 对象。

createChoice 方法同样接收 HttpServletRequest 对象作为参数，通过循环获取多个选项的各个属性，并将它们设置给多个 Choice 对象，最后将这些 Choice 对象添加到一个 List 集合中，并返回该 List 集合。

[48/188] question/QuestionQueryParam.java

这个文件是一个 Java 类文件，文件名是 QuestionQueryParam.java。这个类定义了一个用于存储问题查询参数的对象。它包含三个私有成员变量 assessmentId、status 和 type，以及相应的 getter 和 setter 方法。这个类还实现了一个静态方法 create，用于根据 HttpServletRequest 对象创建一个 QuestionQueryParam 对象，该方法从 HttpServletRequest 对象中获取 assessmentId、status 和 type 参数的值，并将其设置到 QuestionQueryParam 对象中。此外，这个类还重写了 toString 方法，用于将 QuestionQueryParam 对象转化为字符串以便进行打印或者输出。

[49/188] question/SaveServlet.java

该文件是一个名为 SaveServlet.java 的 Java servlet 文件，位于`com.qst.action.question`包下。

该文件实现了一个 Servlet 类，它继承自 BaseServlet 类，并覆盖了 doAction()方法。该方法是 Servlet 的核心方法，用于处理 HTTP 请求和生成 HTTP 响应。它接收从前端页面传递过来

的数据，并保存题目到数据库中。

在文件顶部，通过 `import` 语句引入了一些需要使用的类和接口。这些类和接口包括 `BaseServlet`、`Constant`、`ExamException`、`WebUtil`、`Choice`、`Question`、`User`，以及 `IAssessmentService` 和 `IQuestionService`。

在类中定义了两个私有变量 `assessmentService` 和 `questionService`，它们分别是 `IAssessmentService` 和 `IQuestionService` 的实例。这些服务用于访问和操作题目和评估相关的数据。

在 `doAction()` 方法中，首先通过 `QuestionHelper` 类创建一个 `Question` 对象，并通过 `QuestionHelper` 类创建一个 `Choice` 对象的列表。然后，获取当前用户的信息，并将其设置为题目的用户 ID。接下来，调用 `questionService.save()` 方法保存题目和选项到数据库中。

如果保存成功，将会通过 `addMessage()` 方法添加一条成功消息，并通过 `WebUtil.redirect()` 方法将请求重定向到 `question/dimension.action` 页面，并将题目 ID 作为参数传递过去。

如果保存失败，将会捕获到 `ExamException` 异常。在异常处理代码块中，会通过 `addError()` 方法添加一个错误消息，并将题目和选项设置为请求属性。然后，通过 `WebUtil.forward()` 方法将请求转发到 `question/create.jsp` 页面，以供用户重新创建题目。

总体而言，该文件的功能是接收从前端页面传递过来的题目和选项数据，并保存到数据库中。如果保存成功，将会重定向到题目详情页面；如果保存失败，将会转发到重新创建题目的页面，并显示错误消息。

[50/188] question/UpdateServlet.java

这是一个名为 UpdateServlet.java 的文件，位于 question/目录下。它是一个用于处理题目更新的 Servlet。在 doAction 方法中，它从请求中获取题目和选项的内容，然后根据当前登录用户的信息将题目的修改用户 ID 改为当前登录用户的 ID，调用 QuestionService 中的修改方法来更新题目和选项的内容。接下来，它将消息添加到请求中，重定向到/question/dimension.action 页面，并将题目 ID 作为参数传递给该页面。如果发生任何异常，它将将错误消息添加到请求中，将题目和选项的内容设置为请求属性，然后将请求转发到/question/edit.jsp 页面。

[51/188] question/ViewServlet.java

这是一个名为 ViewServlet.java 的 Java 类文件。它位于 com.qst.action.question 包中。该类是一个 Servlet，它扩展自 BaseServlet 类，并处理/question/view.action 的 HTTP 请求。在 doAction 方法中，该类通过调用其他服务类的方法查找并添加题目、选项、考核类型和问卷维度到 HTTP 请求中，然后将其转发到/question/view.jsp 页面进行显示。

[52/188] reservation/AddServlet.java

这个文件是一个 Java Servlet 文件，它位于 com.qst.action.reservation 包中。它扩展了 BaseServlet 类，并使用@WebServlet 注解将其映射到"/reservation/add.action"路径。
该文件包含一个名为 AddServlet 的类。在该类中，它使用 ServiceFactory 从 IReservationService 接口获取一个实例，用于处理与预约相关的操作。
doAction 方法是该 Servlet 的主要方法。在该方法中，它从请求参数中获取 userId 和 movieId，并调用 reservationService.addReservation 方法将预约添加到数据库中。如果成功，它将在请求中添加一条成功消息，并将请求转发到"/movie/list.action"路径。如果出现异常，它将在请求中添加一条错误消息。
这个 Servlet 主要用于处理添加预约的操作，并提供了对应的请求处理逻辑。

[53/188] reservation/DeleteServlet.java

这个程序文件是`DeleteServlet.java`，位于`reservation`目录下。

这个文件是一个 Servlet 类，继承了`BaseServlet`类。它处理了一个 HTTP POST 请求，映射路径为`/reservation/delete.action`。

在`doAction`方法中，它首先从请求参数中获取`userId`和`movieId`，如果值为 0 则抛出参数错误。

接下来，它调用`reservationService`的`deleteReservation`方法，删除指定的预约。如果删除过程中出现异常，将打印异常信息并将删除失败的错误信息添加到请求中。

最后，如果删除成功，将添加删除成功的消息到请求中，并转发到`/movie/list.action`页面。

注释部分还有一个使用`response.sendRedirect`的语句，不过被注释掉了。

[54/188] reservation/ListServlet.java

该程序文件是一个 Java Servlet，用于处理"/reservation/list.action"的请求。它继承自 `BaseServlet` 类，并实现了 `doAction` 方法来处理具体的业务逻辑。在 `doAction` 方法中，它从请求中获取用户 ID 并调用 `IReservationService` 接口的 `getReservationsByUser` 方法获取该用户的预约列表。然后，将预约列表设置为请求属性，并将请求转发给"/reservation/list.jsp"页面进行显示。

[55/188] schedule/CreateServlet.java

这是一个名为 CreateServlet.java 的程序文件，位于 schedule 目录中。这个文件是一个 Java 类，它继承自 BaseServlet 类，并通过@WebServlet 注解将 URL 映射为"/schedule/create.action"。它实现了一个 doAction 方法，该方法接收 HttpServletRequest 和 HttpServletResponse 对象作为参数，并抛出 ServletException 和 IOException 异常。

在 doAction 方法中，首先获取当前用户的信息，并从请求的属性中获取测试安排对象（如果没有，则创建一个新的测试安排对象）。然后，设置测试安排对象的开始日期和结束日期为当前日期，并将测试安排对象设置为请求的属性。

接下来，通过 ITeamService 和 IAssessmentService 接口获取团队列表和评估类型列表，并将它们设置为请求的属性。

最后，调用 WebUtil 类的 forward 方法将请求转发到"/schedule/create.jsp"页面。

该文件包含了 Schedule 类，User 类，Team 类，AssessmentType 类，IAssessmentService 接口和 ITeamService 接口的导入语句。

[56/188] schedule/DeleteServlet.java

这个程序文件是一个名为 DeleteServlet.java 的 Java 类文件，位于 schedule 目录中。它是一个处理 HTTP POST 请求的 Servlet，用于删除测试安排。在 doAction 方法中，它获取要删除的测试安排的 ID，并通过调用 service 层的方法删除对应 ID 的测试安排。如果在删除过程中发生异常，它会打印异常信息并向请求中添加错误信息。最后，它将重定向到"/schedule/list.action"页面。

[57/188] schedule/EditServlet.java

这是一个 EditServlet.java 文件，位于 schedule 路径下。该文件是一个用于处理编辑行程的

Servlet 类。它继承自 BaseServlet 类，并覆写了 doAction 方法。在 doAction 方法中，它首先获取当前用户信息和请求中的行程 id，并通过调用 Service 类的方法获取对应的行程信息。然后，它获取用户创建的所有团队和所有评估类型的列表，并将它们设置为请求的属性。最后，它将请求转发到/schedule/edit.jsp 页面。

[58/188] schedule/ListServlet.java

这个程序文件是一个 Java 类文件，文件名为 ListServlet.java。它位于 schedule 目录下。该文件是一个 Servlet 类，继承自 BaseServlet 类，用于处理与日程表相关的请求。主要功能是通过 findByCreator 函数查询当前用户创建的日程表数据，并将查询结果存储在请求属性"scheduleList"中，然后将请求转发到/schedule/list.jsp 页面进行展示。

[59/188] schedule/SaveServlet.java

这是一个名为 SaveServlet.java 的 Java 源代码文件。它是一个 Servlet 类，继承自 BaseServlet 类。SaveServlet 类实现了 doAction 方法，该方法在收到 HTTP 请求时会被调用。

在 doAction 方法中，首先创建了一个 Schedule 对象，然后通过 RequestUtil 类从 HTTP 请求中获取一些参数，并将这些参数设置到 Schedule 对象中。接下来设置了一些固定的属性，如 status、creatorId 等，并调用 scheduleService 的 saveSchedule 方法保存了 Schedule 对象。最后，通过 WebUtil 类进行 HTTP 重定向，跳转到了"/schedule/view.action"页面。

如果发生了 ExamException 异常，会打印异常信息，然后调用 addError 方法将异常信息添加到 HTTP 请求的属性中，通过 WebUtil 类进行页面跳转，跳转到"/schedule/create.action"页面。

此文件的主要功能是保存测试安排的信息，并在保存成功后跳转到显示详情页面，同时处理保存失败的情况。

[60/188] schedule/UpdateServlet.java

这是一个名为"UpdateServlet.java"的 Java 程序文件，位于 schedule/目录下。它是一个 Servlet 类，用于处理"/schedule/update.action"的 HTTP 请求。在 doAction 方法中，它首先获取当前用户对象，然后根据请求参数设置了一个 Schedule 对象的各个属性。接着，它调用了一个名为 updateSchedule 的方法来更新 Schedule 对象，并重定向到了"/schedule/view.action?id=" + schedule.getId()的 URL。如果在更新过程中发生了 ExamException 异

常，它会打印异常堆栈跟踪信息，并将异常信息添加到请求属性中，然后将请求转发到"/schedule/update.action"的 URL。

[61/188] schedule/ViewServlet.java

这是一个文件名为 ViewServlet.java 的 Java 源代码文件。代码中定义了一个名为 ViewServlet 的类，该类继承自 BaseServlet。该类是一个 Servlet，它通过处理用户发送的 HTTP 请求来展示一个日程安排。它通过在 doAction 方法中调用 IScheduleService 接口的 findById 方法来获取指定 ID 的日程安排信息，并将其设置到 HttpServletRequest 对象的属性中。最后，它使用 WebUtil 类的 forward 方法将请求转发到/schedule/view.jsp 页面。通过@WebServlet 注解设置了处理该 Servlet 的 URL 为/schedule/view.action。

[62/188] team/CreateServlet.java

这个文件是一个 Java Servlet 程序，命名为 CreateServlet.java。它位于 com.qst.action.team 包中。这个 Servlet 负责处理来自客户端的 HTTP 请求，并在服务器端执行一些操作。在 doAction 方法中，它创建了一个 Team 对象，并设置了一些默认值，然后将这个 Team 对象作为请求属性传递给"/team/create.jsp"页面。最后，它使用 WebUtil 工具类将请求转发到此页面，以便在客户端上显示。整体上，这个 Servlet 用于准备创建班级所需的默认值并在页面上显示。

[63/188] team/DeleteServlet.java

这是一个名为 DeleteServlet.java 的 Java Servlet 文件。它位于 team 目录下。

该 Servlet 接收一个 id 参数，并使用该 id 从数据库中删除一个班级。它实现了 doAction 方法来处理 HTTP 请求。

在 doAction 方法中，首先使用 RequestUtil 工具类从 HttpServletRequest 对象中获取 id 参数的整数值。然后调用 classTeamService 的 deleteTeam 方法来删除数据库中对应的班级。如果出现异常，会将错误信息添加到 HttpServletRequest 对象中。无论是否出现异常，最后都会将请求重定向到/team/list.action 页面。

此文件是一个控制器类，负责接收请求，并调用相应的服务类进行处理。

[64/188] team/EditServlet.java

该文件是一个名为 EditServlet.java 的 Java 源文件，位于 team 目录中。该文件定义了一个名

为 EditServlet 的类，继承自 BaseServlet 类。EditServlet 类是一个 Servlet，用于处理"/team/edit.action"路径的 HTTP 请求。

EditServlet 类包含一个 ITeamService 类型的私有变量 classTeamService，该变量在类的构造函数中通过 ServiceFactory.getService()方法进行初始化。

EditServlet 类重写了 BaseServlet 类的 doAction()方法，该方法处理 HTTP 请求，并根据请求参数中的 id 值查询出要修改的班级，并将其存储到 HttpServletRequest 的"team"属性中。如果查询出现异常，则将异常信息存储到 HttpServletRequest 的错误集合中。

最后，EditServlet 类通过 WebUtil.forward()方法将请求转发到"/team/edit.jsp"页面进行显示。

注释中描述了该文件的作者和用途，即根据 id 参数查询出要修改的班级并显示在页面上。

[65/188] team/ListServlet.java

这个 Java 源代码文件是一个 servlet 文件，用于处理"/team/list.action"请求。它包含一个名为 ListServlet 的类，并继承自 BaseServlet。该类通过获取当前登录用户的信息，并根据用户类型查询班级信息。查询结果会被设置到 HttpServletRequest 对象的属性中，然后通过 WebUtil.forward 方法转发到"/team/list.jsp"页面进行显示。这个类还依赖于 ITeamService 接口和 ServiceFactory 类来实现对班级信息的查询。

[66/188] team/SaveServlet.java

这是一个用于保存团队数据的 servlet 文件。它接收来自前端的表单数据，并将其组装成 Team 对象，然后调用业务方法将 Team 对象保存到数据库中。如果保存成功，它将重定向到团队的详细信息页面，如果保存失败，则在请求属性中添加错误信息，并转发到团队创建页面。

[67/188] team/UpdateServlet.java

这个程序文件是一个名为"UpdateServlet.java"的 Java 代码文件。它位于文件路径"team/"下。

这个文件是一个 Servlet 类，继承自"BaseServlet"类，并使用@WebServlet 注解进行了映射。

这个 Servlet 类的作用是将表单数据更新到数据库，并重定向到"view.action"来显示更新后的数据。

它依赖于以下类和方法：

- BaseServlet 类
- ExamException 类
- RequestUtil 类
- WebUtil 类
- Team 类
- ITeamService 接口
- ServiceFactory 类

在 doAction 方法中，它首先将表单数据封装为 Team 对象，然后调用 ITeamService 接口的 updateTeam 方法来实现更新操作。如果更新失败，则会将错误消息添加到请求中，并将错误消息和 Team 对象传递给"edit.jsp"页面进行显示。

如果更新成功，它会将请求重定向到"/team/view.action?id=" + t.getId()来显示更新后的数据。

[68/188] team/ViewServlet.java

该文件是 Java 代码文件，位于 team 目录下。这个文件定义了一个名为 ViewServlet 的类，继承自 BaseServlet 类。该类使用@WebServlet 注解标注了路径“/team/view.action”，表示当访问该路径时，会通过该 Servlet 来处理请求。

ViewServlet 类中重写了 doAction 方法，该方法接收 HttpServletRequest 和 HttpServletResponse 对象，用来处理前端发送的请求。在该方法中，首先获取要查看的班级的 id，并通过 ITeamService 接口调用 ServiceFactory 获取实现类来查找对应的班级信息。如果查找过程中发生异常，会将异常信息添加到 HttpServletRequest 对象的错误属性中。最后使用 WebUtil.forward 方法，将请求转发到“/team/view.jsp”视图页面进行显示。

[69/188] TestPersonnel/AddServlet.java

这是一个名为 AddServlet.java 的 Java 源代码文件。该文件属于 com.qst.action.TestPersonnel 包，并处理添加测试人员的 HTTP 请求。它从请求中提取信息，执行相应的操作，并根据结果返回成功或错误消息，最后重定向用户到适当的页面。

该文件包含了一些 import 语句，导入了所需的类和接口。它还定义了一个 AddServlet 类，该类继承自 BaseServlet，并被@WebServlet 注解标记为 Servlet，URL 映射路径为"/testPersonnel/add.action"。该类还使用@MultipartConfig 注解来支持文件上传。

在 doAction 方法中，该类从请求中获取用户对象的数据并生成一个 User 对象。然后创建一个 TestPersonnel 对象，并将 User 对象和其他信息填充到 TestPersonnel 对象中。接下来，该类检查手机号是否已经存在，如果存在，则向请求中添加提示并返回到创建页面；否则，调用 service 层方法添加参测人员，并返回成功消息，最后重定向到测试人员列表页面。

该文件还包含了一些私有字段，包括 ITestPersonnelService 和 UserDAO 的实例，用于处理测试人员信息和用户信息的服务和数据访问。

总体来说，该文件实现了处理添加测试人员的 HTTP 请求的功能，并包含了一些必要的导入和字段声明。

[70/188] TestPersonnel/DeleteServlet.java

这个文件是一个 Java Servlet 类，名为 DeleteServlet.java。它位于 TestPersonnel 目录下。

该 Servlet 处理"/testPersonnel/delete.action"请求。在 doAction()方法中，它首先从请求中获取学生的 id，然后调用 ITestPersonnelService 接口的 deleteTestPersonnel()方法删除对应的 TestPersonnel 对象。接着，它使用 WebUtil 工具类将请求转发到"/testPersonnel/list.action?tid=" + testPersonnel.getTeamId()，以显示团队列表页面。

如果删除过程中发生异常，它会捕获 ExamException 并调用 addError()方法将错误信息添加到请求中。然后，它通过 WebUtil 工具类将请求转发到"/testPersonnel/view.action?id=" + id，以显示学生详情页面。

[71/188] TestPersonnel/EditServlet.java

上述代码文件是一个 Java 文件，位于 TestPersonnel 目录下。该文件定义了一个名为 EditServlet 的类，该类继承自 BaseServlet，并使用@WebServlet 注解将 URL 映射到“/testPersonnel/edit.action”。

EditServlet 类中包含一个私有的 ITestPersonnelService 对象 testPersonnelService，以及一个重写了 BaseServlet 的 doAction 方法。在 doAction 方法中，首先通过 RequestUtil.getInt 方法获取请求参数中的"id"值，并传递给 testPersonnelService 的 findById 方法，然后将查询结果存储到 req 的属性"testPersonnel"中，最后通过 WebUtil.forward 方法将请求转发到"/testPersonnel/edit.jsp"页面。

[72/188] TestPersonnel/ImportServlet.java

这个程序文件是一个用于导入参测人员信息的 Servlet 类。它包含以下功能：

1. 在 GET 请求中，将用户重定向到导入页面，用户可以选择要上传的文件。
2. 在 POST 请求中，获取团队 ID 和上传的文件。
3. 创建一个文件 reader 并逐行读取上传的文件。
4. 每行使用逗号分隔数据，如果数据格式不正确，则添加错误消息并重定向到导入页面。
5. 创建 TestPersonnel 和 User 对象，设置相应的属性。
6. 将 TestPersonnel 对象添加到列表中，以便进一步处理。
7. 调用服务层方法将数据导入到数据库。
8. 添加成功消息并重定向到参测人员列表页面，如果导入过程中出现异常，则添加错误消息并重定向到团队列表页面。

该 Servlet 类的映射地址为"/testPersonnel/import.action"，并使用了 MultipartConfig 注解来处理文件上传。

[73/188] TestPersonnel/ListServlet.java

这个文件是一个 Java Servlet 文件，位于`com.qst.action.TestPersonnel`包中的`ListServlet.java`文件。

该 Servlet 的主要功能是获取与特定班级相关的测试人员列表以及所有的团队列表，并将这些数据设置到请求属性中，以便在 JSP 页面中使用和显示。

Servlet 类标记为`@WebServlet("/testPersonnel/list.action")`，表示该 Servlet 将处理来自`/testPersonnel/list.action`路径的 HTTP 请求。

该文件还声明了两个成员变量`classTeamService`和`testPersonnelService`，用于获取班级团队和测试人员的相关数据。

`doAction`方法是 Servlet 的主要执行方法，它从 HTTP 请求中获取班级 ID 并使用`testPersonnelService`的`findByTeamId`方法获取与该班级相关的测试人员列表。然后，将测试人员列表和团队列表设置为请求属性，并将请求转发到`/testPersonnel/list.jsp`页面。

整体而言，这个 Servlet 的功能是获取与特定班级相关的测试人员列表和所有团队列表，并将其传递给 JSP 页面进行显示。

[74/188] TestPersonnel/SelectServlet.java

这段代码是一个 Servlet 类，它是一个 Java 类，用于处理 HTTP 请求和响应。具体来说，它是一个用于查询测试人员信息的 Servlet。

这个 Servlet 类包含一个 doAction 方法，它是 Servlet 的主要方法，用于处理请求和生成响应。在这个方法中，首先从请求参数中获取班级 id、姓名和手机号，然后调用测试人员服务的 query 方法查询符合条件的测试人员信息。接着，将查询结果和所有的班级列表设置为请求属性，以供后续的 JSP 页面使用和显示。最后，通过 WebUtil 类的 forward 方法将请求转发到"/testPersonnel/list.jsp"页面。

此外，这个 Servlet 类还定义了两个成员变量 classTeamService 和 testPersonnelService，它们都是通过 ServiceFactory 获取的服务类的实例。

总之，这个 Servlet 类的功能是接收并处理查询测试人员信息的 HTTP 请求，并将结果返回给客户端。

[75/188] TestPersonnel/UpdateServlet.java

这是一个名为 UpdateServlet.java 的 Java 文件，位于 TestPersonnel 目录下。它是一个 Servlet 类，用于处理从前端接收到的请求并更新测试人员信息。

该文件包含了一系列 import 语句，用于导入所需的类和接口。主要导入了 com.qst 包下的一些类和接口，以及 java.io.IOException、javax.servlet.ServletException、javax.servlet.annotation.WebServlet 和 javax.servlet.http 包下的一些类。

该类继承自 BaseServlet，覆盖了父类的 doAction 方法。在 doAction 方法中，首先创建了一个 User 对象，并从请求中获取并设置其 id、name 和 status 属性。然后创建了一个 TestPersonnel 对象，并设置其 id、user、birthDate、phone 和 gender 属性。接着通过 ITestPersonnelService 接口的实现类 ServiceFactory.getService 方法获取一个测试人员服务对象。调用 testPersonnelService 的 updateTestPersonnel 方法更新测试人员信息。

try-catch 块用于捕捉可能抛出的 ExamException 异常。如果发生异常，将其打印出来并将错误消息添加到请求中。然后将 testPersonnel 对象设置为请求属性，并将请求转发到/testPersonnel/edit.jsp 页面。如果没有发生异常，将请求重定向到/testPersonnel/view.action 页面，并附带上更新后的测试人员 id 作为参数。

这是一个用于处理更新测试人员信息的 Servlet 类。

[76/188] TestPersonnel/ViewServlet.java

这是一个名为"ViewServlet.java"的 Java 代码文件，位于 TestPersonnel 路径下。

该文件中定义了一个名为"ViewServlet"的 servlet 类，继承自"BaseServlet"类。
在"ViewServlet"类中，重写了"doAction"方法，该方法处理 HTTP 请求并执行相应的操作。

在"doAction"方法中，首先通过"RequestUtil.getInt"方法获取 HTTP 请求中的参数"id"的整数值。然后使用获取到的 id 值通过"testPersonnelService"对象调用"findById"方法查询数据库，获取对应 id 的"TestPersonnel"对象。

接着，将查询到的"TestPersonnel"对象设置为"req"的属性，并将请求转发到"/testPersonnel/view.jsp"页面。

另外，该文件还导入了一些其他的类，包括"BaseServlet"、"RequestUtil"、"WebUtil"等，用于提供相关辅助方法。还导入了"ITestPersonnelService"和"ServiceFactory"这两个接口和类，用于获取"testPersonnelService"对象。

总的来说，该文件是一个用于展示"TestPersonnel"的详细信息的 servlet 类。通过接收 HTTP 请求中的参数"id"，查询数据库获取对应 id 的"TestPersonnel"对象，并将该对象绑定到请求属性中，最后转发到一个 JSP 页面进行展示。

[77/188] user/CreateServlet.java

这个程序文件是一个新建用户的 Servlet 类。它继承了`BaseServlet`类，重写了`doAction`方法。在该方法中，创建了一个新的`User`对象，并设置了状态和类型。然后，将用户对象设置为请求的属性，并通过`WebUtil`工具类转发到"/user/create.jsp"页面。该 Servlet 的 URL 映射为"/user/create.action"。

[78/188] user/DeleteServlet.java

这是一个 Java Servlet 类，文件名为 DeleteServlet.java。它位于 com.qst.action.user 包下，继承自 BaseServlet 类。该类使用@WebServlet 注解将路径"/user/delete.action"映射到该 Servlet。

该 Servlet 类实现了 doAction 方法，该方法处理用户发起的 HTTP 请求，并执行以下操作：

1. 从 HttpServletRequest 对象中获取名为"id"的参数，并将其转为整数。

2. 调用 userAdminService 的 deleteUserById 方法，通过 id 删除用户。
3. 使用 WebUtil 类的 redirect 方法将响应重定向到"/user/list.action"路径。

该 Servlet 类有一个名为 userAdminService 的私有成员变量，该变量的类型是 IUserAdminService 接口，通过 ServiceFactory 的 getService 方法获取实例。

总结：DeleteServlet 类是一个处理用户删除操作的 Servlet，它接收用户请求，通过 id 删除用户，并将响应重定向到用户列表页面。

[79/188] user/EditServlet.java

该程序文件是一个名为 EditServlet.java 的 Servlet 类。它位于 com.qst.action.user 包下。该类继承自 BaseServlet，并使用@WebServlet 注解进行映射，将路径"/user/edit.action"映射到 doAction()方法上。

在 doAction()方法中，该类通过 RequestUtil.getInt()方法从请求中获取 id 参数的整数值，并调用 userAdminService 的 findUserById()方法来查找对应 id 的用户对象。然后，将查找到的用户对象存储到请求属性"user"中。最后，通过 WebUtil.forward()方法将请求转发到"/user/edit.jsp"页面。

该类的主要功能是处理用户编辑操作的请求，并将对应的用户信息展示到编辑页面上。

[80/188] user/ListServlet.java

这个程序文件是一个名为 ListServlet.java 的 Java 类文件，位于 user/目录下。它是一个 Servlet 类，继承自 BaseServlet 类，并且被@WebServlet 注解标记为处理"/user/list.action"请求的 Servlet。
在 doAction 方法中，它调用了一个叫做 findUsers 的方法来获取用户列表数据，并将数据存储在名为"userList"的请求属性中。然后使用 WebUtil 工具类将请求转发到"/user/list.jsp"页面进行展示。

[81/188] user/PasswordServlet.java

这个程序文件是一个名为 PasswordServlet.java 的 Java Servlet 类文件，位于 com.qst.action.user 包下。该类是一个密码管理的 Servlet，用于重置用户密码。它继承自 BaseServlet，并使用@WebServlet 注解来指定访问路径为"/user/password.action"。

在 doAction 方法中，首先通过 RequestUtil.getInt 方法获取请求参数中的 id，然后调用

userAdminService 的 resetPassword 方法重置对应 id 的用户密码。接着通过 addMessage 方法向请求中添加一条消息"密码已重置"。最后调用 WebUtil 的 forward 方法将请求转发到"/user/list.action"路径。

此文件依赖于 com.qst 包下的其他类，并通过 ServiceFactory 获取了一个 IUserAdminService 的实例来实现具体的密码重置操作。

[82/188] user/RegServlet.java

这个文件是一个 Java 源代码文件，位于“user/”目录下，文件名为“RegServlet.java”。它是一个用于处理用户注册请求的 Servlet 类。在该类中，会通过 HTTP 请求获取用户的一些信息，然后将这些信息保存到数据库中。如果用户已经被注册过了，则会给用户一个注册失败的提示；如果注册成功，则会将用户的信息保存到 Session 中，并重定向到登录页面。

[83/188] user/SaveServlet.java

该文件是一个 Java 源代码文件，位于 user/目录下。该文件包含一个名为 SaveServlet 的 Java 类，用于处理保存用户信息的请求。该类继承自 BaseServlet 类，并通过@WebServlet 注解映射了 URL 路径/user/save.action。

在 doAction 方法中，该类根据请求中的参数创建 User 对象，并调用 userAdminService 的 saveUser 方法保存用户信息。如果保存成功，通过 addMessage 方法将消息添加到请求中，并通过 WebUtil 类的 redirect 方法重定向到用户信息查看页面。如果保存失败，捕获 ExamException 异常，将用户对象和错误信息信息添加到请求中，并通过 WebUtil 类的 forward 方法转发到用户信息创建页面。

该类依赖于 com.qst 包中的其他类，包括 BaseServlet、ExamException、RequestUtil、WebUtil、User、IUserAdminService 和 ServiceFactory。其中，IUserAdminService 和 ServiceFactory 通过 ServiceFactory.getService 方法获取实例。

[84/188] user/ShowRegServlet.java

这是一个用于显示注册页面的 Java Servlet 文件。它位于`com.qst.action.user`包中。该 Servlet 使用了`BaseServlet`类的继承，并实现了`doAction`方法，该方法处理 HTTP 的 GET 请求。Servlet 的 URL 映射为`/user/showReg.do`。

在`doAction`方法中，它首先调用`classTeamService`的`findAll`方法获取所有的`Team`对象，并将

其设置为请求的属性 `teamList`。然后，它使用 `WebUtil.forward` 方法将请求和响应转发到 `/user/reg.jsp` 页面。

如果发生 `ExamException` 异常，它将调用 `addError` 方法将错误信息设置为请求的属性。但是，关于此部分的注释代码被注释掉了，所以实际上不会执行。

总体上，这个 Servlet 的目的是用于显示注册页面，并获取相关的数据。

[85/188] user/UpdateServlet.java

这个文件是一个 Java 源代码文件，它位于 user/UpdateServlet.java。代码文件实现了一个名为 UpdateServlet 的类，继承了 BaseServlet 类。该类是一个 Servlet，处理用户更新操作。在 doAction 方法中，通过 HttpServletRequest 对象获取用户更新的信息，然后调用 userAdminService 的 updateUser 方法将更新后的用户信息保存到数据库中。如果更新出现异常，会将错误信息返回给前端界面展示。

[86/188] user/ViewServlet.java

这个程序文件名为 ViewServlet.java，位于 user/目录下。它是一个用于处理用户查看操作的 Servlet。这个 Servlet 继承了 BaseServlet 类，并实现了 doAction 方法，该方法用于处理 HTTP 请求。在 doAction 方法中，它首先获取请求参数中的 id，然后通过 userAdminService 调用 findUserById 方法查找对应 id 的用户信息，并将用户信息存放在请求属性中。最后，通过 WebUtil 类的 forward 方法将请求转发给/user/view.jsp 页面进行展示。

[87/188] dao/AssessmentTypeDAO.java

该文件是一个名为 AssessmentTypeDAO.java 的 Java 类文件，它位于 dao/目录下。这个类是一个数据访问对象（DAO），用于与数据库交互，并提供了一些关于考核类型（AssessmentType）的操作方法。其中包括了查询所有考核类型、插入新的考核类型、根据 ID 删除考核类型、根据名称查询考核类型、根据 ID 查询考核类型和更新考核类型等方法。这些方法通过执行 SQL 语句与数据库进行交互，并将结果封装成 AssessmentType 对象返回。在这个类中还定义了一个私有方法 createAssessmentType 用于将数据库查询结果转换为 AssessmentType 对象。整个类的目的是提供对考核类型数据的访问和操作功能。

[88/188] dao/ChoiceDAO.java

该程序文件是一个名为 ChoiceDAO.java 的 Java 类文件。它位于 dao/目录下。

该类包含以下公共方法：

- deleteByQuestion(int id)：根据问题 ID 删除选项
- update(Choice ch)：更新选项
- insert(Choice ch)：插入新的选项
- findByQuestion(Integer id)：根据问题 ID 查找选项

该类还包含一个私有方法：

- create(ResultSet rs)：根据结果集创建一个 Choice 对象

以上方法使用了与数据库交互的代码，并通过 Db 类获取数据库连接进行操作。在数据库操作过程中，如果发生异常，将抛出 ExamException 异常。

此外，该类还导入了 com.qst.entity.Choice 类和其他类文件，这些类文件在此代码中未显示。

总体来说，该类是一个用于与数据库交互的数据访问对象（DAO），主要用于对选项进行增删改查的操作。

[89/188] dao/DAOFactory.java

这个程序文件是一个 DAO 工厂类，用于获取各种类型的数据访问对象（DAO）。它使用了一个 Map 来存储不同类型的 DAO，并在静态代码块中进行初始化。通过调用 `getDAO` 方法并传入一个 DAO 类的 Class 对象，可以获取该对应类型的 DAO 实例。

[90/188] dao/DimensionDAO.java

这是一个名为 DimensionDAO.java 的文件，位于 com.qst.dao 包下。这个文件包含了对问卷维度数据访问对象的定义。主要包括以下方法：

1. findByAssessment(int assessmentId)：根据考核类型 ID 从数据库查询问卷维度列表，并将每个问卷维度存储到一个 List 集合中。
2. findById(int id)：通过问卷维度 ID 在数据库中查询问卷维度信息。
3. findByAssessmentAndTitle(int assessmentId, String title)：通过考核类型 ID 和问卷维度标题查询是否有相同的问卷维度。
4. insert(PersonalityDimension kp)：向数据库中插入问卷维度。
5. update(PersonalityDimension kp)：更新数据库中的问卷维度信息。
6. findDimensionByQuestion(int questionId)：根据问题 ID 查询相关的问卷维度列

表。
7. delete(int id)：根据问卷维度 ID 删除数据库中的问卷维度。

这个文件还包含了一个私有方法 create(ResultSet rs)，用于从数据库结果集中创建 PersonalityDimension 对象。

以上是文件的主要内容和功能概述。

[91/188] dao/ExamDAO.java

这是一个名为 ExamDAO.java 的文件，它是一个用于访问数据库的 DAO（数据访问对象）类。该类定义了一些方法来执行对数据库中"exams"表的操作，包括查找、插入和更新。这些方法使用预编译的 SQL 语句执行数据库查询和更新操作。文件中还包含一个私有方法用于从结果集中创建 Exam 对象。该类还导入了其他类（com.qst.Db 和 com.qst.entity.Exam）来完成数据库连接和 Exam 对象的处理。

[92/188] dao/ExamQuestionDAO.java

这是一个 Java 类文件，文件名为 ExamQuestionDAO.java。该类位于 com.qst.dao 包下。该类主要包含两个方法：

1. save 方法：用于将 ExamQuestion 对象保存到数据库中。它接收一个 ExamQuestion 对象作为参数，并把对象的各个属性值插入到 exam_questions 表中。该方法抛出 SQLException 异常。
2. deleteByExam 方法：用于根据指定的 examId，从数据库中删除相应的记录。它接收一个整型参数 examId，并执行相应的数据库操作。该方法抛出 SQLException 异常。

该类还依赖于其他类和接口，其中 Db 类用于获取数据库连接，ExamQuestion 类用于表示考试问题的实体。

[93/188] dao/MovieDAO.java

该文件是一个名为 MovieDAO.java 的 Java 类文件。它位于 dao 目录下。该类用于与数据库进行交互，提供了以下功能：

- createMovie()：根据数据库返回的 ResultSet 创建一个 Movie 对象。
- getAllMovies()：获取所有电影信息，并返回一个包含 Movie 对象的 List。

- getMovieById(int id)：根据电影 ID 获取电影信息，并返回一个 Movie 对象。
- addMovie(Movie movie)：向数据库中添加电影信息。
- updateMovie(Movie movie)：更新数据库中的电影信息。
- deleteMovie(int id)：根据电影 ID 从数据库中删除电影信息。
- reserve(int id, int num)：预约电影的座位，更新数据库中的预约座位数。

该类还使用了其他类和接口：

- com.qst.Db：用于获取数据库连接。
- com.qst.ExamException：自定义异常类。
- com.qst.SMMSUploader：一个用于上传图片的工具类。
- com.qst.entity.Movie：电影实体类，用于存储电影信息。

通过调用这些方法可以实现对电影信息的增删改查操作，并且可以进行电影座位的预约。

[94/188] dao/QuestionDAO.java

这个程序文件是一个 Java 类，文件名为 QuestionDAO.java。它在 com.qst.dao 包下。该类提供了与数据库交互的方法，用于查询、插入、更新和删除问题。它还包括一些辅助方法，如查找与问题关联的维度和创建问题对象。该类使用了 Db 类来获取数据库连接，并使用 ExamException 类来处理异常。

[95/188] dao/ReservationDAO.java

这个程序文件是一个 Java 类文件，名为 ReservationDAO.java。它位于 dao 文件夹下。

该类包含以下方法：

- createReservation：根据传入的 ResultSet 对象创建并返回一个 Reservation 对象。
- getReservationsByUser：根据用户 ID 获取该用户的所有预订记录，并将其封装在一个 List对象中返回。
- getReservationsByMovie：根据电影 ID 获取该电影的所有预订记录，并将其封装在一个 List对象中返回。
- addReservation：将一个 Reservation 对象添加到数据库中。
- deleteReservation：从数据库中删除指定用户 ID 和电影 ID 的预订记录。
- isReserved：检查指定用户 ID 和电影 ID 的预订记录是否存在于数据库中。

该类依赖于 com.qst.Db 和 com.qst.entity.Reservation 类。它使用 Db 类获取数据库连接，并操作数据库执行相关查询和更新操作。

值得注意的是，一些方法在处理数据库异常时会打印异常堆栈信息，而其他方法则会将异常转换为自定义的 ExamException 异常并抛出。

该类主要用于与数据库交互，提供了对预订记录的创建、获取、添加、删除和检查等操作。

[96/188] dao/ScheduleDAO.java

这个程序文件是一个名为 ScheduleDAO.java 的 Java 类文件。它位于`com.qst.dao`包下。该类提供了与调度（schedule）相关的数据库操作方法。它主要包括以下方法：

1. `findByCreatorId(Integer id)`：通过创建者的 ID 查询调度信息。
2. `findById(Integer id)`：通过调度的 ID 查询调度信息。
3. `save(Schedule h)`：保存调度信息到数据库。
4. `update(Schedule h)`：更新数据库中的调度信息。
5. `delete(Integer id)`：从数据库中删除指定的调度。
6. `findByTeamId(int teamId)`：通过团队的 ID 查询团队的调度信息。

此外，该类还包括一个私有方法`create(ResultSet rs)`，用于从查询结果集中创建调度对象。

[97/188] dao/TeamDAO.java

这个文件是一个 Java 类文件，文件名是 TeamDAO.java。它位于 dao/目录下。

该类是 Team（班级）的数据库访问对象，实现了对班级数据表的增删改查操作。具体包括以下方法：

- `findAll()`：查询所有班级并返回一个 Team 对象的列表。
- `findByCreator(Integer creator_id)`：根据创建者 id 查询对应的班级，并返回一个 Team 对象的列表。
- `findById(Integer id)`：根据班级 id 查询对应的班级，并返回一个 Team 对象。
- `save(Team t)`：将一个 Team 对象保存（插入）到班级数据表中。
- `update(Team t)`：更新班级数据表中指定 id 对应的班级信息。
- `delete(int id)`：从班级数据表中删除指定 id 对应的班级。

此外，还有一个私有方法`create(ResultSet rs)`，用于将查询结果集的一行数据封装为一个

Team 对象并返回。

该类所在的包路径是 com.qst.dao。

[98/188] dao/TestPersonnelDAO.java

这是一个名为 TestPersonnelDAO.java 的源代码文件。它属于 com.qst.dao 包，并包含了一些与测试人员数据访问相关的方法。该文件定义了以下方法：

- findByTeam(int id)：根据给定的团队 ID 从数据库中查询相关的测试人员信息，并将这些信息以列表的形式返回。
- query(int teamId, String name, String stdNo)：根据给定的班级 ID、姓名和学号从数据库中查询相关的测试人员信息，并将这些信息以列表的形式返回。
- findById(int id)：根据给定的测试人员 ID 从数据库中查询对应的测试人员信息并返回。
- update(TestPersonnel s)：根据给定的测试人员对象更新数据库中对应的测试人员信息。
- findByphone(String phone)：根据给定的电话号码从数据库中查询对应的测试人员信息并返回。
- insert(TestPersonnel s)：将给定的测试人员对象插入到数据库中。
- delete(int id)：根据给定的测试人员 ID 从数据库中删除对应的测试人员信息。

此外，该文件还定义了私有方法 create(ResultSet rs)，用于根据 ResultSet 对象创建 TestPersonnel 对象，以及一个静态常量字符串 select，用于存储一个 SQL 查询语句。

[99/188] dao/UserDAO.java

这个程序文件是一个 Java 类文件，命名为 UserDAO.java。该文件定义了一个名为 UserDAO 的类，这个类主要用于操作对应的数据库表 users。

这个类中包含了以下方法：

- findAll(): 查询并返回所有用户记录的列表。
- findByLogin(String login): 根据登录名查询并返回对应的用户记录。
- findById(int id): 根据用户 ID 查询并返回对应的用户记录。
- insert(User u): 将一个新的用户记录插入到数据库表中。
- update(User u): 更新指定用户的信息。
- updatePassword(int userId, String password): 更新指定用户的密码。

- delete(int userId): 删除指定用户的记录。
- updateLastLogin(Integer userId, Timestamp lastLogin): 更新指定用户的最后登录时间信息。
- createUser(ResultSet rs): 根据数据库查询结果创建并返回一个用户对象。

这个类使用了其他类和接口，包括 Db 类（用于获取数据库连接和执行 SQL 操作），ExamException 类（自定义的异常类，用于处理 SQL 异常），User 类（用于表示用户实体）。

[100/188] entity/AssessmentType.java

这是一个名为 AssessmentType.java 的 Java 源代码文件。它位于 entity/目录下。该文件定义了一个继承自 BaseEntity 的 AssessmentType 类，表示考核类型。

AssessmentType 类具有以下成员变量：

- title：考核类型的标题
- cost：考核类型的费用
- status：考核类型的状态

AssessmentType 类还有以下方法：

- getTitle()：获取考核类型的标题
- setTitle()：设置考核类型的标题
- getCost()：获取考核类型的费用
- setCost()：设置考核类型的费用
- getStatus()：获取考核类型的状态
- setStatus()：设置考核类型的状态

该文件的主要功能是定义和访问考核类型的相关属性。

[101/188] entity/BaseEntity.java

这是一个名为"BaseEntity"的 Java 类文件。它是一个实体类，实现了 Serializable 接口。该类具有以下主要功能：

1. 该类包含一个 Integer 类型的 id 属性，表示实体的唯一标识符。
2. 它包含一个"extras"字段，它是一个 HashMap 对象，用于在对象之间传递数据。

3. 类中还有一个静态的"desc"字段，它是一个 HashMap 对象，用于将代号映射到描述信息。
4. 该类定义了一个静态的"getDesc"方法，用于根据给定的类、前缀和代码返回描述信息。
5. 它还定义了一个静态的"addDesc"方法，用于将一对类、描述信息添加到描述映射中。
6. 该类提供了用于获取和设置 id 属性以及获取 extras 属性的 getter 和 setter 方法。

[102/188] entity/Choice.java

这是一个 Java 文件，文件名是 Choice.java。它位于 entity/路径下。该文件定义了一个名为 Choice 的类，这个类继承自 BaseEntity 类。Choice 类包含了私有的实例变量 questionId、title、hint 和 checked，分别表示问题的 ID、标题、提示和是否被选中。该类还提供了相应的 getter 和 setter 方法用于访问和设置这些实例变量的值。

[103/188] entity/Exam.java

该文件是一个 Exam 类的源代码文件，位于 entity/目录下。该类继承自 BaseEntity 类，并具有以下属性：

- testPersonnelId: Integer 类型，试题人员 ID
- scheduleId: Integer 类型，试题计划 ID
- beginTime: Timestamp 类型，试题开始时间
- endTime: Timestamp 类型，试题结束时间
- schedule: Schedule 类型，试题计划
- examQuestions: List类型，考试试题列表，包括试题，学生答案等信息
- result: String 类型，考试结果
- questionIndex: int 类型，当前试题索引

该类还提供了以下方法：

- getQuestion(): 返回当前试题的 ExamQuestion 对象
- getPassedTime(): 返回已考时间（单位：分钟）
- getQuestionIndex(): 返回当前试题索引
- setQuestionIndex(int questionIndex): 设置试题索引
- getTestPersonnelId(): 返回试题人员 ID

- getExamQuestions(): 返回考试试题列表
- setExamQuestions(List examQuestions): 设置考试试题列表
- setTestPersonnelId(Integer testPersonnelId): 设置试题人员 ID
- getScheduleId(): 返回试题计划 ID
- setScheduleId(Integer scheduleId): 设置试题计划 ID
- getBeginTime(): 返回试题开始时间
- setBeginTime(Timestamp beginTime): 设置试题开始时间
- getEndTime(): 返回试题结束时间
- setEndTime(Timestamp endTime): 设置试题结束时间
- getResult(): 返回考试结果
- setResult(String result): 设置考试结果
- getSchedule(): 返回试题计划对象
- setSchedule(Schedule schedule): 设置试题计划对象
- toString(): 返回对象的字符串表示形式

[104/188] entity/ExamQuestion.java

该文件是一个 Java 类文件，名为 `ExamQuestion`。该类包含以下成员变量：

- `examId`：整数类型，表示考试 ID。
- `testPersonnelId`：整数类型，表示考试人员 ID。
- `questionId`：整数类型，表示题目 ID。
- `answer`：整型数组，表示学生的答题结果，不是试题的真正答案。
- `right`：布尔类型，表示学生的答题是否正确。
- `score`：整数类型，表示本题分数。
- `question`：Question 类型，表示题目对象。

类中包含以下方法：

- `getQuestion()`：获取题目对象。
- `setQuestion(Question question)`：设置题目对象。
- `getExamId()`：获取考试 ID。
- `setExamId(Integer examId)`：设置考试 ID。
- `getTestPersonnelId()`：获取考试人员 ID。
- `setTestPersonnelId(Integer testPersonnelId)`：设置考试人员 ID。
- `getQuestionId()`：获取题目 ID。

- `setQuestionId(Integer questionId)`：设置题目 ID。
- `getAnswer()`：获取学生答题结果。
- `setAnswer(int[] answer)`：设置学生答题结果。
- `isRight()`：判断学生答题是否正确。
- `setRight(boolean right)`：设置学生答题是否正确。
- `getScore()`：获取本题分数。
- `setScore(int score)`：设置本题分数。

[105/188] entity/Movie.java

这个程序文件名为 Movie.java，位于 entity/路径下。它是一个实体类，继承自 BaseEntity 类。该类有以下属性：

- title: 电影标题，字符型；
- description: 电影描述，字符型；
- date: 电影日期，Timestamp 型；
- capacity: 电影座位容量，整型；
- reservedSeats: 电影预留座位数，整型；
- posterUrl: 电影海报 URL，字符型。

该类提供了默认构造函数和带参数的构造函数用于初始化属性值。另外，还提供了各属性的 getter 和 setter 方法，并重写了 toString 方法以便打印 Movie 对象的信息。

[106/188] entity/PersonalityDimension.java

这个程序文件是一个 Java 类文件，文件名是 PersonalityDimension.java。该文件位于 entity 目录下。这个类继承自 BaseEntity 类，拥有四个成员变量：title（String 类型）、depict（String 类型）、assessmentId（int 类型）。该类还有对应这四个成员变量的 getter 和 setter 方法。

[107/188] entity/Question.java

这是一个 Question（问题）的实体类。它有以下属性：

- userId（用户 ID）：表示问题所属的用户 ID。
- assessmentId（评估 ID）：表示问题所属的评估 ID。
- title（标题）：表示问题的标题。

- hint（提示）：表示问题的提示。
- type（类型）：表示问题的类型。
- difficulty（难度）：表示问题的难度。
- status（状态）：表示问题的状态。
- choices（选项）：表示问题的选项列表。

该实体类还有以下方法：

- isAnswerRight（判断答案是否正确）：根据传入的答案数组，判断给定的答案是否正确。

[108/188] entity/Reservation.java

这个文件是一个名为 Reservation 的实体类，它继承了 BaseEntity 类。它有两个私有属性 userId 和 movieId，以及默认的构造函数和带参数的构造函数。它还重写了 toString 方法，并且提供了获取和设置 userId 和 movieId 的方法。

[109/188] entity/Schedule.java

这个 Java 文件是一个名为 Schedule 的实体类。它具有以下属性：

- beginDate：开始日期
- endDate：结束日期
- duration：持续时间
- status：状态
- assessmentId：评估 ID
- teamId：团队 ID
- creatorId：创建者 ID
- createDate：创建日期
- team：团队对象
- creator：创建者对象
- assessmentType：评估类型对象
- questionNumber：问题数量

该类还有一些其他方法，包括获取和设置属性的方法。它还实现了一个静态块，用于添加描述信息，并提供了一个 getStatusDesc()方法来获取状态的描述。

[110/188] entity/Team.java

这个程序文件是一个 Java 类文件，名字是"Team.java"，位于"com.qst.entity"包下。这个类继承自"BaseEntity"类，并且定义了一些属性和对应的 get/set 方法。这些属性包括：name（String 类型）、beginYear（Date 类型）、status（int 类型）和 creatorId（int 类型）。

[111/188] entity/TestPersonnel.java

这是一个名为`TestPersonnel.java`的 Java 程序文件。它位于`entity/`目录下。

该文件定义了一个`TestPersonnel`类，它继承自`BaseEntity`类。`TestPersonnel`类具有以下属性：

- `id`：一个整数，表示测试人员的唯一标识符。
- `phone`：一个字符串，表示测试人员的电话号码。
- `gender`：一个字符串，表示测试人员的性别。
- `birthDate`：一个`java.sql.Date`对象，表示测试人员的出生日期。
- `teamId`：一个整数，表示测试人员所属的团队的标识符。
- `user`：一个`User`对象，表示测试人员的用户信息。

该类还实现了一些方法：

- `getUser()`：返回`user`属性的值。
- `setUser(User user)`：设置`user`属性的值。
- `getTeamId()`：返回`teamId`属性的值。
- `setTeamId(int teamId)`：设置`teamId`属性的值。
- `getId()`：重写了`BaseEntity`类中的`getId()`方法，返回`id`属性的值。
- `setId(Integer id)`：重写了`BaseEntity`类中的`setId(Integer id)`方法，设置`id`属性的值。
- `getPhone()`：返回`phone`属性的值。
- `setPhone(String phone)`：设置`phone`属性的值。
- `getGender()`：返回`gender`属性的值。
- `setGender(String gender)`：设置`gender`属性的值。
- `getBirthDate()`：返回`birthDate`属性的值。
- `setBirthDate(Date birthDate)`：设置`birthDate`属性的值。

[112/188] entity/User.java

这段代码是一个 Java 类文件，文件名为 User.java。该类是一个实体类，用于表示用户信息。它继承了一个名为 BaseEntity 的类。

代码中定义了一些属性，如登录名（login）、姓名（name）、密码（passwd）、用户类型（type）、用户状态（status）和最后登录时间（lastLogin）。还定义了一些方法，如获取和设置属性值的方法，获取用户类型和用户状态的描述信息的方法。

代码中还有一些静态代码块和静态方法，用于添加用户状态和用户类型的描述信息。

总结起来，这个类用于表示用户对象的信息，包括基本属性和对属性进行操作的方法。

[113/188] service/IAssessmentService.java

这是一个 Java 源代码文件，文件名是 IAssessmentService.java。该文件定义了一个接口 IAssessmentService，包含以下方法：

1. findAllAssessment：返回一个 AssessmentType 对象列表，表示找到所有的考核类型。
2. findAssessmentById：根据给定的 id 参数，返回对应的 AssessmentType 对象，表示找到特定 id 的考核类型。
3. updateAssessment：根据给定的 assessmentType 参数，更新考核类型的信息。
4. saveAssessment：根据给定的 assessmentType 参数，保存考核类型。
5. deleteAssessment：根据给定的 id 参数，删除对应的考核类型。

[114/188] service/IDimensionService.java

这是一个名为 IDimensionService.java 的接口文件。它定义了一些操作维度（dimension）的方法。这些方法包括：通过测量 ID 查找维度、通过 ID 查找维度、保存维度、更新维度以及删除维度。

[115/188] service/IExamService.java

这是一个名为 IExamService 的接口文件，位于 com.qst.service 包中。它定义了一系列与考试相关的方法，包括：

- findScheduleByTestPersonnel：根据 TestPersonnel 参数查询该学生的所有考试

安排（已考和未考的）
- begin：开始一场考试，参数包括 TestPersonnel 和 Schedule 的 id
- end：结束一场考试，参数为 Exam 对象
- examResult：根据 id 查询考试的结果，可能会抛出 SQLException 异常。

接口中还有一个注释块，当前注释块中没有任何内容。

[116/188] service/IMovieService.java

这个程序文件是一个接口文件，命名为 IMovieService.java。这个接口定义了一些与电影相关的操作方法：

- `getAllMovies()`：获取所有电影的列表。
- `getMovieById(int id)`：根据电影 ID 获取电影。
- `addMovie(Movie movie)`：向电影列表中添加电影。
- `updateMovie(Movie movie)`：更新电影信息。
- `deleteMovie(int id)`：根据电影 ID 删除电影。

这些方法提供了对电影数据的基本操作，可以在实现类中具体实现。

[117/188] service/IQuestionService.java

这个程序文件是一个接口文件，命名为"IQuestionService.java"。它定义了一个名为 IQuestionService 的接口。该接口定义了一些方法来处理问卷调查相关的操作，包括查询问题、查询选项、查询问卷维度、保存问题、附加维度、更新问题以及删除问题等。

[118/188] service/IReservationService.java

这个程序文件是接口文件，命名为 IReservationService.java。它位于 service 目录下。该接口定义了用于处理预订操作的方法。其中包括获取某个用户的预订列表、获取某个电影的预订列表、检查某个用户是否已经预订过某个电影、添加预订和删除预订等方法。可以看出，这个接口主要是为了与预订相关的业务逻辑。

[119/188] service/IScheduleService.java

这个程序文件是一个 Java 接口文件，文件名是 IScheduleService.java。这个接口定义了一些与日程相关的操作方法:

- `List<Schedule> findByCreator(User user)`：根据用户查找创建者的所有日程
- `Schedule findById(Integer id)`：根据 ID 查找日程
- `void saveSchedule(Schedule h)`：保存日程
- `void updateSchedule(Schedule h)`：更新日程
- `Schedule deleteSchedule(Integer id)`：删除日程

这些方法封装了与日程相关的数据操作，可以被其他类实现来实现具体的逻辑。

[120/188] service/ITeamService.java

这是一个 Java 接口文件，文件名为 ITeamService.java。该接口定义了一些班级操作的方法，包括 findAll()、findByCreator(Integer creator_id)、findById(Integer id)、saveTeam(Team t)、updateTeam(Team t)和 deleteTeam(int id)等。这些方法提供了对班级信息的增删改查操作。接口中还包含了一些注释，用于说明方法的参数和用途。

[121/188] service/ITestPersonnelService.java

这是一个名为 ITestPersonnelService 的接口文件，位于 com.qst.service 包下。它定义了一些与测试人员相关的操作。包括根据团队 ID 查询人员、根据 ID 查询人员、导入测试人员、更新测试人员、添加测试人员、根据团队 ID、姓名、学号查询人员、删除测试人员等方法。该接口的实现类可以通过编写相应的代码来完成具体的功能。

[122/188] service/IUserAdminService.java

该文件是一个 Java 接口文件，位于`service`路径下。文件定义了一个名为`IUserAdminService`的接口，该接口包含以下方法：

- `List<User> findUsers()`：用于查询用户列表。
- `User findUserById(int id)`：用于根据用户 ID 查询用户。
- `void saveUser(User u)`：用于保存用户。
- `void updateUser(User user)`：用于更新用户信息。
- `void deleteUserById(int id)`：用于根据用户 ID 删除用户。
- `void resetPassword(int id)`：用于重置用户密码。

[123/188] service/IUserService.java

这是一个名为 IUserService.java 的程序文件，位于 service 目录下。该文件定义了一个接口

IUserService，包含了以下方法：

- saveUser: 保存用户信息
- login: 用户登录，传入登录名和密码，返回 User 对象
- changePassword: 修改用户密码，传入用户 ID、旧密码和新密码。

这个接口提供了一些用户相关的操作方法，但是注释掉了其他方法，可能是因为暂时不需要或者还未实现。这个接口可能是为了用户管理和认证功能而设计的。

[124/188] service/ServiceFactory.java

这是一个名为 ServiceFactory 的 Java 类文件。它是一个工厂类，提供了获取各种服务实例的方法。

该类包含一个静态的 Map 对象 services，用于存储各种服务接口与其对应的实现类实例。

在静态代码块中，将各种服务接口与其对应的实现类实例放入 services Map 中。

该类还提供了一个用于获取服务实例的静态方法 getService，接受一个 Class 对象作为参数，并使用 clazz 作为 key 从 services Map 中获取对应的实例。

可以通过调用 getService 方法来获取具体的服务实例。

[125/188] service/impl/AssessmentServiceImpl.java

这是一个名为 AssessmentServiceImpl.java 的 Java 类文件。该文件位于 service/impl/路径下。该文件是一个实现了 IAssessmentService 接口的类，主要用于处理考核类型相关的业务逻辑。

该文件包含以下主要功能：

- findAll()方法：在数据库中查询所有考核类型信息，并返回列表。
- createAssessmentType()方法：解析结果集并创建一个 AssessmentType 对象。
- findById(int id)方法：根据给定的 id 在数据库中查询并返回对应的考核类型。
- findAllAssessment()方法：查询并返回所有考核类型。
- findAssessmentById(int id)方法：根据给定的 id 查询并返回对应的考核类型。
- updateAssessment(AssessmentType assessment)方法：根据给定的考核类型信息更新数据库中的考核类型。
- saveAssessment(AssessmentType assessment)方法：将给定的考核类型信息保

存到数据库中。
- deleteAssessment(int id)方法：删除指定 id 的考核类型，但前提是该考核类型没有关联的试题。

该文件中还使用了以下 DAO 接口：

- AssessmentTypeDAO：用于操作考核类型相关的数据访问对象。
- QuestionDAO：用于操作试题相关的数据访问对象。

总的来说，该文件实现了 IAssessmentService 接口，并封装了与考核类型相关的业务逻辑，包括查询、更新、保存和删除等功能。

[126/188] service/impl/DimensionServiceImpl.java

这是一个名为 DimensionServiceImpl.java 的类文件，位于 service/impl/路径下。这个类实现了 IDimensionService 接口，并提供了一些维度相关的功能。具体包括根据考核 ID 查找维度、根据 ID 查找维度、保存维度、更新维度和删除维度等功能。文件中还引入了一些依赖，并使用了 DAOFactory 来获取维度和问题的 DAO 实例。

[127/188] service/impl/ExamServiceImpl.java

这个文件是一个 Java 类文件，它位于 service/impl/ExamServiceImpl.java。它是一个名为 ExamServiceImpl 的实现了 IExamService 接口的类。

该类包含了一些属性，如 ExamDAO、ScheduleDAO、AssessmentTypeDAO、ExamQuestionDAO 和 QuestionService，这些属性在类的构造函数中初始化。

该类实现了 IExamService 接口的方法，包括 findScheduleByTestPersonnel、begin、end 等。这些方法用于从数据库中查询考试安排、开始考试和结束考试，并进行相关的处理和操作。

它还包含了一些私有方法，如 findQuestion 和 isCanExam，这些方法用于在考试过程中执行一些辅助功能，例如查找考试题目和判断考生是否可以参加考试等。

总体上看，这个类实现了一个考试服务的业务逻辑，包括查询考试安排、开始考试、判卷和结束考试等功能。它依赖于其他的 DAO 和 Service 组件来完成具体的数据操作。

[128/188] service/impl/MovieServiceImpl.java

该程序文件是一个 Java 类文件，名为 MovieServiceImpl.java，位于 service/impl/路径下，

它实现了 IMovieService 接口，并提供了对电影数据的增删改查功能。这些功能是通过 MovieDAO 类来实现的，该类的实例被创建并存储在 MovieServiceImpl 类中的 movieDAO 成员变量中。该类包含以下方法：

- getAllMovies(): 返回所有电影的列表。
- getMovieById(int id): 根据电影的 id 返回对应的电影对象。
- addMovie(Movie movie): 添加新的电影。
- updateMovie(Movie movie): 更新已有电影的信息。
- deleteMovie(int id): 根据电影的 id 删除对应的电影。

[129/188] service/impl/QuestionServiceImpl.java

这个文件是一个 Java 类文件，文件名是 QuestionServiceImpl.java。它位于 service/impl/目录下。

这个类实现了 IQuestionService 接口，并包含了多个方法用于处理问题及选项的操作。其中包括根据参数查找问题、根据 ID 查找问题、根据问题 ID 查找选项、根据问题 ID 查找问卷维度、保存问题及选项、更新问题及选项、删除问题及选项等功能。

该类还包含了一些私有方法用于校验选项的有效性，并提供了一些方法用于题目与问卷维度的关联操作，以及查询问题与问卷维度关联关系的方法。

该类依赖于其他类，包括 ChoiceDAO、DimensionDAO 和 QuestionDAO，并且通过 DAOFactory 获取 DAO 实例。

总的来说，这个类提供了一些与问题及选项相关的功能，并与数据库进行交互。

[130/188] service/impl/ReservationServiceImpl.java

这个程序文件是一个服务实现类，文件路径是`service/impl/ReservationServiceImpl.java`。该类实现了`IReservationService`接口，并包含以下方法：

- `getReservationsByUser(int user)`：根据用户 ID 获取预订列表。
- `getReservationsByMovie(int movie)`：根据电影 ID 获取预订列表。
- `isReserved(int userId, int movieId)`：检查给定用户和电影是否已被预订。
- `addReservation(int userId, int movieId)`：添加一条新的预订记录。
- `deleteReservation(int user, int movie)`：删除指定的预订记录。

该类还包含一个私有的`ReservationDAO`对象，用于与数据库进行交互。

[131/188] service/impl/ScheduleServiceImpl.java

这个源代码文件是实现了一个名为 ScheduleServiceImpl 的类，实现了 IScheduleService 接口。

这个类包含了一些私有的成员变量，包括 ScheduleDAO、UserDAO、TestPersonnelDAO、AssessmentTypeDAO 和 TeamDAO，它们都是通过 DAOFactory 获取的。

这个类实现了 IScheduleService 接口中定义的方法，包括 findByCreator、findById、saveSchedule、updateSchedule 和 deleteSchedule。

在 findByCreator 方法中，通过调用 scheduleDAO 的方法来获取一组 Schedule 对象，并对每个对象的 creator、assessmentType 和 team 进行了设置。

在 findById 方法中，通过调用 scheduleDAO 的方法来获取一个指定 id 的 Schedule 对象，并对对象的 creator、assessmentType 和 team 进行了设置。

在 saveSchedule 方法中，通过调用 scheduleDAO 的方法来保存一个 Schedule 对象。

在 updateSchedule 方法中，通过调用 scheduleDAO 的方法来更新一个 Schedule 对象。

在 deleteSchedule 方法中，通过调用 scheduleDAO 的方法来删除一个指定 id 的 Schedule 对象，并在对象的状态不是未开始或作废时抛出异常。

整体而言，这个类用于处理与考试安排相关的业务逻辑。

[132/188] service/impl/TeamServiceImpl.java

这是一个文件名为 TeamServiceImpl.java 的 Java 源代码文件。它位于 service/impl/目录下。该文件定义了一个名为 TeamServiceImpl 的类，实现了 ITeamService 接口。该类包含了对 Team 实体对象的数据库操作的具体实现，例如根据创建者 ID 查找团队、查找所有团队、根据 ID 查找团队、保存团队、更新团队以及删除团队等方法。在异常情况下，会抛出 ExamException 异常。

[133/188] service/impl/TestPersonnelServiceImpl.java

这个程序文件是一个 Java 类文件，文件名是 TestPersonnelServiceImpl.java。它位于

service/impl/目录下。该文件实现了 ITestPersonnelService 接口，并提供了测试人员信息的相关业务逻辑。它包括了通过团队 ID 查找相关的测试人员、根据 ID 查找测试人员、查询参测人员信息、更新测试人员信息、导入测试人员信息、添加测试人员、删除测试人员等方法。这些方法主要是通过调用 TestPersonnelDAO 和 UserDAO 中的方法对数据库进行操作，同时处理了可能出现的 SQLException 异常，并抛出自定义的 ExamException 异常。

[134/188] service/impl/UserAdminServiceImpl.java

这是一个名为 UserAdminServiceImpl.java 的 Java 类文件，属于 service/impl 目录下的文件。该文件实现了 IUserAdminService 接口，并提供了用户管理的相关功能，包括查询用户、根据用户 ID 查找用户、保存用户、更新用户、删除用户和重置密码。具体实现了接口定义的方法，并使用了 UserDAO 类来访问数据库进行操作。

[135/188] service/impl/UserServiceImpl.java

这个文件是`UserServiceImpl.java`，位于`service/impl/`目录下。它是一个 Java 源代码文件，实现了`IUserService`接口。该类提供了用户管理的功能，包括用户登录、新增用户、修改密码等。

[185/188] mbti.zip.extract/mbti/pom.xml

这是一个名为"MBTI"的项目的 Maven 配置文件（pom.xml）。它使用 war 打包类型，并具有一些依赖项，包括"jstl"、"mysql-connector-java"、"javax.servlet-api"、"hanlp"、"unirest-java"和"jackson-databind"。此外，该文件包含一些插件配置，如"maven-clean-plugin"、"maven-resources-plugin"、"maven-compiler-plugin"、"maven-surefire-plugin"、"maven-war-plugin"、"maven-install-plugin"和"maven-deploy-plugin"等。项目的版本号为 1.0-SNAPSHOT。

[186/188] mbti.zip.extract/mbti/src/main/webapp/WEB-INF/web.xml

这个程序文件是一个 Java Web 应用程序的配置文件(web.xml)。它位于 mbti.zip.extract/mbti/src/main/webapp/WEB-INF 目录下。该配置文件定义了应用程序的相关配置，包括应用程序的展示名称和默认的欢迎页面(login.jsp)。

[187/188] mbti.zip.extract/mbti/target/MBTI/WEB-INF/web.xml

这个程序文件是一个标准的 JavaEE Web 应用的配置文件（web.xml）。它的作用是定义了 Web 应用的基本信息，包括显示名称和欢迎页面。在该文件中，应用的显示名称为"exam4"，而 login.jsp 被指定为默认的欢迎文件。

文件名	功能
BaseFilter.java	定义了一个抽象过滤器类，用于 Servlet 请求
BaseServlet.java	定义了一个抽象 Servlet 类，提供一些通用的
Constant.java	包含了一些常量值，用于项目中的常量定义
Db.java	提供了数据库连接和操作的工具类
EncodeFilter.java	一个过滤器类，用于设置请求和响应的字符编
EntityToMapConverter.java	提供了将实体对象转换为 Map 对象的工具方
ExamException.java	定义了一个自定义异常类，用于处理与考试相
MysqlCjAbandonedConnectionCleanupListener.java	监听器类，用于在 ServletContext 初始化和销毁时进行一些操作
package-info.java	包级别的注释文件，用于定义 com.qst 包的
RequestUtil.java	提供了一些工具方法，用于处理 HttpServle 对象的一些操作
SecurityFilter.java	一个过滤器类，用于对指定 URL 模式的请求
SMMSUploader.java	提供了上传文件到 SM.MS 图片托管服务的功
Test.java	一个用于测试的类，打印出 uri 中斜杠"/"的
WebUtil.java	包含了一些静态方法，用于在 Web 应用程序中进行请求转发和重定向操作
LoginServlet.java	处理用户登录请求，并进行相关的验证和处理
LogoutServlet.java	处理用户登出请求的 Servlet

文件名	功能
	类，使当前会话失效，重定向到登录页面
PasswordServlet.java	处理用户密码相关的功能，包括修改密码和
BaiduAIAPI.java	调用百度 AI 接口生成文字的 API。
ChatServlet.java	处理聊天的请求，调用 BaiduAIAPI 进行文字生成，并提取关键词转换为图像。
TextRankKeyword.java	提取关键词的工具类，使用 TextRank 算法。
WordToImageConverter.java	实现文字转图像的 API。
CreateServlet.java	处理创建考核类型的请求。
DeleteServlet.java	处理删除考核类型的请求。
EditServlet.java	处理编辑考核类型的请求。
ListServlet.java	处理显示考核类型列表的请求。
SaveServlet.java	处理保存考核类型信息的请求。
UpdateServlet.java	处理更新考核类型信息的请求。
ViewServlet.java	处理查看单个考核类型的请求。
BeginServlet.java	处理开始考试的请求。
EndServlet.java	处理结束考试的请求。
ExamServlet.java	处理考试相关的请求。
exam/ResultServlet.java	处理获取考试结果的请求
movie/AddServlet.java	处理添加电影的请求
movie/DeleteServlet.java	处理删除电影的请求
movie/EditServlet.java	处理编辑电影的请求
movie/ListServlet.java	处理获取电影列表的请求

文件名	功能
PersonalityDimension/DeleteServlet.java	处理删除问卷维度的请求
PersonalityDimension/EditServlet.java	处理编辑问卷维度的请求
PersonalityDimension/ListServlet.java	处理获取问卷维度列表的请求
PersonalityDimension/SaveServlet.java	处理保存问卷维度的请求
PersonalityDimension/UpdateServlet.java	处理更新问卷维度的请求
PersonalityDimension/ViewServlet.java	处理查看问卷维度详情的请求
question/DeleteServlet.java	处理删除题目的请求
question/DimensionServlet.java	处理维度相关操作的请求
question/EditServlet.java	处理修改题目的请求
question/ListServlet.java	处理获取题目列表的请求
question/QuestionHelper.java	提供辅助方法来处理获取题目数据的请求
question/QuestionQueryParam.java	定义了问题查询参数的对象
question/SaveServlet.java	处理题目保存的 Servlet
question/UpdateServlet.java	处理题目更新的 Servlet
question/ViewServlet.java	处理题目展示的 Servlet
reservation/AddServlet.java	处理预订添加的 Servlet
reservation/DeleteServlet.java	处理预订删除的 Servlet
reservation/ListServlet.java	处理预订列表的 Servlet
schedule/CreateServlet.java	处理日程创建的 Servlet
schedule/DeleteServlet.java	处理日程删除的 Servlet
schedule/EditServlet.java	处理日程编辑的 Servlet
schedule/ListServlet.java	处理日程列表的 Servlet

文件名	功能
schedule/SaveServlet.java	处理日程保存的 Servlet
schedule/UpdateServlet.java	处理日程更新的 Servlet
schedule/ViewServlet.java	处理日程展示的 Servlet
team/CreateServlet.java	处理班级创建的 Servlet
team/DeleteServlet.java	处理班级删除的 Servlet
team/EditServlet.java	处理团队编辑请求的 Servlet
team/ListServlet.java	获取班级和团队列表的 Servlet
team/SaveServlet.java	保存团队数据的 Servlet
team/UpdateServlet.java	更新团队数据的 Servlet
team/ViewServlet.java	查看团队详情的 Servlet
TestPersonnel/AddServlet.java	添加测试人员信息的 Servlet
TestPersonnel/DeleteServlet.java	删除测试人员信息的 Servlet
TestPersonnel/EditServlet.java	编辑测试人员信息的 Servlet
TestPersonnel/ImportServlet.java	导入测试人员信息的 Servlet
TestPersonnel/ListServlet.java	获取测试人员列表的 Servlet
TestPersonnel/SelectServlet.java	查询测试人员信息的 Servlet
TestPersonnel/UpdateServlet.java	更新测试人员信息的 Servlet
TestPersonnel/ViewServlet.java	查看测试人员详情的 Servlet
user/CreateServlet.java	创建用户的 Servlet
user/DeleteServlet.java	删除用户的 Servlet
user/EditServlet.java	编辑用户的 Servlet
ListServlet.java	获取用户列表数据

文件名	功能
PasswordServlet.java	重置用户密码
RegServlet.java	处理用户注册请求
SaveServlet.java	保存用户信息
ShowRegServlet.java	显示注册页面
UpdateServlet.java	处理用户更新操作
ViewServlet.java	查看用户信息
AssessmentTypeDAO.java	对考核类型的数据库操作
ChoiceDAO.java	对选项的数据库操作
DAOFactory.java	获取各种类型的数据访问对象
DimensionDAO.java	对问卷维度的数据库操作
ExamDAO.java	对考试的数据库操作
ExamQuestionDAO.java	对考题的数据库操作
MovieDAO.java	对电影信息的数据库操作
QuestionDAO.java	对问题的数据库操作
ReservationDAO.java	对预订记录的数据库操作
ScheduleDAO.java	提供与调度（schedule）相关的数据库操作
TeamDAO.java	提供对班级数据表的增删改查操作。
TestPersonnelDAO.java	提供与测试人员数据访问相关的方法。
UserDAO.java	提供对用户数据表的增删改查操作。
AssessmentType.java	定义了考核类型的属性和方法。
BaseEntity.java	定义了实体类的基本属性和方法。
Choice.java	定义了选项的属性。

文件名	功能
Exam.java	定义了考试的属性。
ExamQuestion.java	定义了考试问题的属性。
Movie.java	定义了电影的属性。
PersonalityDimension.java	定义了人格维度的属性。
Question.java	定义了问题的属性。
Reservation.java	定义了预约的属性。
Schedule.java	定义了调度的属性。
Team.java	定义了班级的属性。
TestPersonnel.java	定义了测试人员的属性。
entity/User.java	实现用户信息的表示
service/IAssessmentService.java	处理考核类型相关的业务逻辑
service/IDimensionService.java	处理维度相关的操作
service/IExamService.java	实现考试相关的业务逻辑
service/IMovieService.java	处理电影相关的操作
service/IQuestionService.java	处理问卷调查相关的操作
service/IReservationService.java	实现预订相关的业务逻辑
service/IScheduleService.java	处理日程相关的操作
service/ITeamService.java	处理班级相关的操作
service/ITestPersonnelService.java	处理测试人员相关的操作
service/IUserAdminService.java	处理用户管理相关的操作
service/IUserService.java	处理用户相关的操作
service/ServiceFactory.java	提供各种服务实例的获取方法

文件名	功能
service/impl/AssessmentServiceImpl.java	实现了 IAssessmentService 接口的考核类型
service/impl/DimensionServiceImpl.java	实现了 IDimensionService 接口的维度相关
service/impl/ExamServiceImpl.java	实现了 IExamService 接口的考试相关的业务
service/impl/MovieServiceImpl.java	实现了对电影数据的增删改查功能
service/impl/QuestionServiceImpl.java	处理问题和选项的操作
service/impl/ReservationServiceImpl.java	提供预定服务相关的业务逻辑
service/impl/ScheduleServiceImpl.java	提供对日程安排的操作和管理
service/impl/TeamServiceImpl.java	实现了团队相关功能，包括团队的创建、查找
service/impl/TestPersonnelServiceImpl.java	实现了对测试人员信息的操作和管理
service/impl/UserAdminServiceImpl.java	实现对用户的管理，包括用户查询、更新、修
service/impl/UserServiceImpl.java	提供用户登录、新增用户和修改密码等用户管
pom.xml	Maven 项目的配置文件，定义了项目的依赖
web.xml (src/main/webapp/WEB-INF/web.xml)	Web 应用程序的配置文件，定义了应用程序
web.xml (MBTI/WEB-INF/web.xml)	JavaEE Web 应用程序的配置文件，定义了应

这些文件是一个基于 MBTI 的 Web 应用程序，实现了用户登录、登出和相关功能，包含了数据库操作、请求过滤、异常处理等辅助功能。具有用户登录、登出功能，提供了考核类型的管理和考试功能，并提供了聊天功能和文字转图像功能。并提供了电影的增删改查功能，以及对问卷维度和题目的操作。考核类型的管理和考试功能，以及电影、问卷、题目的增删改查操作。