Part IX
Case Studies

Part IX presents a case study that uses multiple Java EE technologies.

This part contains the following chapter:

Duke's Tutoring Case Study Example

The Duke's Tutoring example application is a tracking system for a tutoring center for students.

Students or their guardians can check in and out, the tutoring center can track attendance and status updates, and store contact information for guardians and students.

The following topics are addressed here:

- Design and Architecture of Duke's Tutoring
- Main Interface
- Administration Interface
- Running the Duke's Tutoring Case Study Application

Design and Architecture of Duke's Tutoring

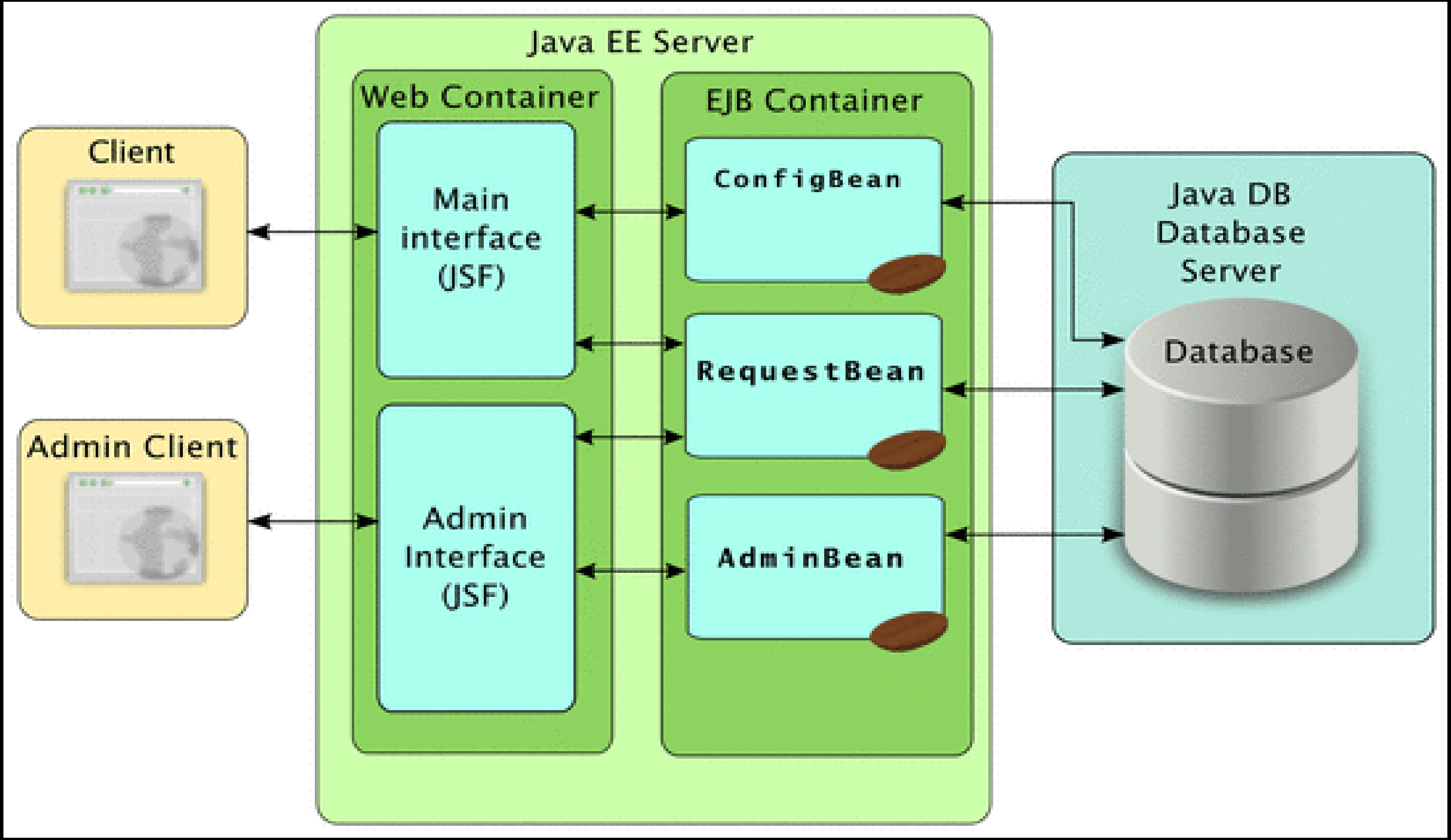
Duke's Tutoring is a web application that incorporates several Java EE technologies.

It exposes both a main interface (for students and guardians) and an administration interface (for tutoring center staff to maintain the system).

The business logic for both interfaces is provided by enterprise beans.

The enterprise beans use the Java Persistence API to create and store the application's data in the database.

Figure 49-1 Architecture of the Duke's Tutoring Example Application.



Duke's Tutoring uses the following Java EE 6 platform features:

- Java Persistence API entities
  - Java API for JavaBeans Validation (Bean Validation) annotations on the entities for verifying data
  - A custom Bean Validation annotation, `@Email`, for validating email addresses

- Enterprise beans
    - Local, no-interface view session and singleton beans
    - JAX-RS resources in a session bean
    - Java EE security constraints on the administrative interface business methods
    - All enterprise beans packaged within the WAR

- JavaServer Faces technology, using Facelets for the web front-end
  - Templating
  - Composite components
  - A custom formatter, `PhoneNumberFormatter`
  - Security constraints on the administrative interface
  - AJAX-enabled Facelets components

The Duke's Tutoring application has two main user interfaces, both packaged within a single WAR file:

- The main interface, for students, guardians, and staff
- The administrative interface used by the staff to manage the students and guardians, and to generate attendance reports

Apart from the main and administrative interface, there is a JUnit test that demonstrates how to use the embedded EJB container to test the business logic of the session beans.

Main Interface

The main interface allows students and staff to check students in and out, and record when students are outside at the playground.

Java Persistence API Entities Used in the Main Interface

The entities used in the main interface encapsulate data stored and manipulated by Duke's Tutoring, and are located in the `dukestutoring.entity` package.

The `Person` entity defines attributes common to students and guardians tracked by the Duke's Tutoring application.

These attributes are the person's name and contact information, including phone numbers and email address.

The phone number and email address attributes have Bean Validation annotations to ensure that the submitted data is well-formed.

The email attribute uses a custom validation class, `dukestutoring.util.Email`.

`Person` has two subclasses, `Student` and `Guardian`.

For additional data common to all people, the `PersonDetails` entity is used to store attributes like pictures and the person's birthday that aren't included in the `Person` entity for performance reasons.

The `Student` entity stores attributes specific to the students that come to tutoring.

This includes information like the student's grade level and school.

The `Guardian` entity's attributes are specific to the parents or guardians of a `Student`.

Students and guardians have a many-to-many relationship.

That is, a student may have a one or more guardians, and a guardian may have one or more students.

The `Address` entity represents a mailing address, and is associated with `Person` entities.

Addresses and people have a many-to-one relationship.

That is, one person may have many addresses.

The `TutoringSession` entity represents a particular day at the tutoring center.

A particular tutoring session tracks which students attended that day, and which students went to the park.

Associated with `TutoringSession` is the `StatusEntry` entity, which logs when a student's status changes.

Student's status changes when they check in to a tutoring session, when they go to the park, and when they check out.

The status entry allows the tutoring center staff to track exactly which students attended a tutoring session, when they checked in and out, which students went to the park while they were at the tutoring center, and when they went to and came back from the park.

For information on creating Java Persistence API entities, see Chapter 32, Introduction to the Java Persistence API.

For information on validating entity data, seeValidating Persistent Fields and Properties and Chapter 47, Advanced Bean Validation Concepts and Examples.

Enterprise Beans Used in the Main Interface

The enterprise beans used in the main interface provide the business logic for Duke's Tutoring, and are located in the `dukestutoring.ejb` package.

`ConfigBean` is singleton session bean used to create the default students when the application is initially deployed, and to create an automatic EJB timer that creates tutoring session entities every weekday.

`RequestBean` is a stateless session bean containing the business methods for the main interface.

Students or staff can check students in and out and track when they go to and come back from the park.

The bean also has business methods for retrieving lists of students.

The business methods in `RequestBean` use strongly-typed Criteria API queries to retrieve data from the database.

For information on creating and using enterprise beans, see Part IV, Enterprise Beans.

For information on creating strongly-typed Criteria API queries, see Chapter 35, Using the Criteria API to Create Queries.

Facelets Files Used in the Main Interface

The Duke's Tutoring application uses Facelets to display the user interface, and makes extensive use of the templating features of Facelets.

Facelets is the default display technology for JavaServer Faces, and consists of XHTML files located in the *tut-install*`/examples/case-studies/dukes-tutoring/web` directory.

The following Facelets files are used in the main interface:

`template.xhtml`

Template file for the main interface.

`error.xhtml`

Error file if something goes wrong (this shouldn't occur).

`index.xhtml`

Landing page for the main interface.

`park.xhtml`

Page showing who is currently at the park.

`current.xhtml`

Page showing who is currently in today's tutoring session.

`statusEntries.xhtml`

Page showing the detailed status entry log for today's session.

`resources/components/allStudentsTable.xhtml`

A composite component for a table displaying all active students.

`resources/components/currentSessionTable.xhtml`

A composite component for a table displaying all students in today's session.

`resources/components/parkTable.xhtml`

A composite component for a table displaying all students currently at the park.

`WEB-INF/includes/navigation.xhtml`

XHTML fragment for the main interface's navigation bar.

`WEB-INF/includes/footer.xhtml`

XHTML fragment for the main interface's footer.

For information on using Facelets, see Chapter 5, Introduction to Facelets.

Helper Classes Used in the Main Interface

The following helper classes, in the `dukestutoring.util` package, are used in the main interface:

`CalendarUtil`

A class that provides a method to strip the unnecessary time data from `java.util.Calendar` instances.

`Email`

Custom Bean Validation annotation class for validating email addresses in the `Person` entity.

`StatusType`

An enumerated type defining the different statuses that a student can have.

Possible statuses are: `IN`, `OUT`, and `PARK`.

`StatusType` is used throughout the application, including in the `StatusEntry` entity, and throughout the main interface.

`StatusType` also defines a `toString` method that returns a localized translation of the status based on the locale.

Properties Files

The strings used in the main interface are encapsulated into resource bundles to allow the display of localized strings in multiple locales.

Each of the following properties files has locale-specific files, appended with the locale code, containing the translated strings for that locale.

For example, `Messages_es.properties` contains the localized strings for Spanish locales.

`ValidationMessages.properties`

Strings for the default locale used by the Bean Validation runtime to display validation messages.

This file must be named `ValidationMessages.properties` and located in the default package as required by the Bean Validation specification.

`dukestutoring/web/messages/Messages.properties`

Strings for the default locale for the main and administration Facelets interface.

For information on localizing web applications, see Registering Custom Localized Static Text and Registering Custom Error Messages.

Deployment Descriptors Used in Duke's Tutoring

The following deployment descriptors are used in Duke's Tutoring:

`src/conf/beans.xml`

An empty deployment descriptor file used to enable the CDI runtime.

`web/WEB-INF/faces-config.xml`

The JavaServer Faces configuration file.

`src/conf/persistence.xml`

The Java Persistence API configuration file.

`web/WEB-INF/glassfish-web.xml`

The GlassFish-specific configuration file.

web/WEB-INF/web.xml

The web application configuration file.

No enterprise bean deployment descriptor is used in Duke's Tutoring.

Annotations in the enterprise bean class files are used for the configuration of enterprise beans in this application.

Administration Interface

The administration interface of Duke's Tutoring is used by the tutoring center staff to manage the data used by the main interface: the students, the student's guardians, and the addresses.

The administration interface uses many of the same components as the main interface.

Additional components that are only used in the administration interface are described here.

Enterprise Beans Used in the Administration Interface

The following enterprise beans, in the `dukestutoring.ejb` package, are used in the administration interface:

AdminBean

A stateless session bean for all the business logic used in the administration interface.

Contains security constraint annotations to allow invocation of the business methods only by authorized users.

Facelets Files Used in the Administration Interface

The following Facelets files are used in the administration interface:

`admin/adminTemplate.xhtml`

Template for the administration interface.

`admin/index.xhtml`

Landing page for the administration interface.

`admin/login.xhtml`

Login page for the security-constrained administration interface.

`admin/loginError.xhtml`

Page displayed if there are errors authenticating the administration user.

`admin/address` directory

Pages that allow you to create, edit, and delete `Address` entities.

`admin/guardian` directory

Pages that allow you to create, edit, and delete `Guardian` entities.

`admin/student` directory

Pages that allow you to create, edit, and delete `Student` entities.

`resources/components/formLogin.xhtml`

Composite component for a login form using Java EE security.

WEB-INF/includes/adminNav.xhtml

XHTML fragment for the administration interface's navigation bar.

Running the Duke's Tutoring Case Study Application

This section describes how to build, package, deploy, and run the Duke's Tutoring application.

Setting Up GlassFish Server

Before running the Duke's Tutoring application, set up the security realm used by Duke's Tutoring with users and groups.

The user names and passwords set in this security realm are used to log in to the administration interface of Duke's Tutoring.

Duke's Tutoring maps users in the `admins` group of the server's security realm to the `Administrator` role used in the security constraint annotations in `AdminBean`.

Adding a User to the File Realm in GlassFish Server

Use the Administration Console of GlassFish Server to add a new user to the `file` security realm.

Before You Begin

Make sure GlassFish Server is started as described in Starting and Stopping the GlassFish Server.

1. Open a web browser to the Administration Console at `http://localhost:4848`.
2. In the left pane, expand Configurations → server-config → Security → Realms, and click “file.”
3. On the Edit Realm page, click Manage Users.

4. On the File Users page, click New.
5. In the User ID field, type a user name under User ID.
6. In the Group List field, type `admins`.
7. In the New Password and Confirm New Password fields, type a password.
8. Click OK.

Running Duke's Tutoring

This section describes how to build, deploy, and run Duke's Tutoring in NetBeans IDE and using Ant.

To Build and Deploy Duke's Tutoring in NetBeans IDE

Before You Begin

You must have already configured GlassFish Server as a Java EE server in NetBeans IDE, as described in To Add GlassFish Server as a Server in NetBeans IDE.

1. From the File menu, choose Open Project.
2. In the Open Project dialog, navigate to:

`tut-install/examples/case-studies/`

3. Select the `dukes-tutoring` folder.
4. Select the Open as Main Project check box and the Open Required Projects check box.
5. Click Open Project.

Note - The first time you open Duke's Tutoring in NetBeans, you will see error glyphs in the project pane.

This is expected, as the metamodel files used by the enterprise beans for Criteria API queries have not yet been generated.

6. Right-click `dukes-tutoring` in the project pane and select Run.

This will build, package, and deploy Duke's Tutoring to GlassFish Server, starting the Java DB database and GlassFish Server if they've not already been started.

After the application has been successfully deployed, the Duke's Tutoring main interface will open in a web browser if NetBeans IDE has been configured to open web applications in a web browser.

To Build and Deploy Duke's Tutoring Using Ant

Before You Begin

Make sure GlassFish Server is started as described in Starting and Stopping the GlassFish Server, and Java DB server is started as described in Starting and Stopping the Java DB Server.

1. In a terminal window, go to:

```
tut-install/examples/
case-studies/dukes-tutoring/
```

2. Type the following command:

```
ant all
```

This command builds, packages, and deploys Duke's Tutoring to GlassFish Server.

Using Duke's Tutoring

Once Duke's Tutoring is running on GlassFish Server, use the main interface to experiment with checking students in and out or sending them to the park.

Using the Main Interface of Duke's Tutoring

1. In a web browser, open the main interface at the following URL:

```
http://localhost:8080/
dukes-tutoring/
```

2. **Use the main interface to check students in and out, and to log when the students go to the park.**

Using the Administration Interface of Duke's Tutoring

Follow these instructions to log in to the administration interface of Duke's Tutoring and add new students, guardians, and addresses.

1. In a web browser, open the administration interface at the following URL:

```
http://localhost:8080/
dukes-tutoring/admin/index.xhtml
```

This will redirect you to the login page.

2. At the login page, enter the username and password you configured inAdding a User to the File Realm in GlassFish Server .

3. Use the administration interface to add or modify students, guardians, or addresses.