

Welcome to this lecture! Today, you'll learn about deploying your web information systems and you'll have another opportunity to put your knowledge into practice by means of a simple project.

--

Copyright (C) 2018 Universidad de Sevilla

The use of these slides is hereby constrained to the conditions of the TDG Licence, a copy of which you may download from <http://www.tdg-seville.info/License.html>

The problem: a short-message log



We Chirp

*The world's favourite
short-message log!*

The problem that we're going to solve in this lecture is about a very simple short-message log to which we refer to as "We chirp".

Ready to start riding?



This problem's like riding this bicycle. It's a little more difficult than the previous problems because you have to pay attention to a variety of details concerning converters, repositories, services, and controllers, but it's not very difficult.

You're not totally constrained!



As usual, we provide you with a harness, that is, a template project. Contrarily to the previous projects, you're not totally constrained because the project that we provide has a lot of holes, which means that you have to make a lot of decisions. But you're still constrained since you can't change the template that we provide. Please, search for "TODO:" to find out what you have to do.

This might be an interview test



This is not the kind of problem that you're going to solve on your first day as a professional software engineer. Remember that you're not likely to work on more than a repository, a service, a view or a controller on your first working day; and typically, someone else will provide you with the other pieces of the project. But it is very likely what you might be requested to produce a project like this in a serious interview. If you plan on earning more than 500€ per month, then get ready for an interview in which you have to prove that you can implement a simple project like "We Chirp".

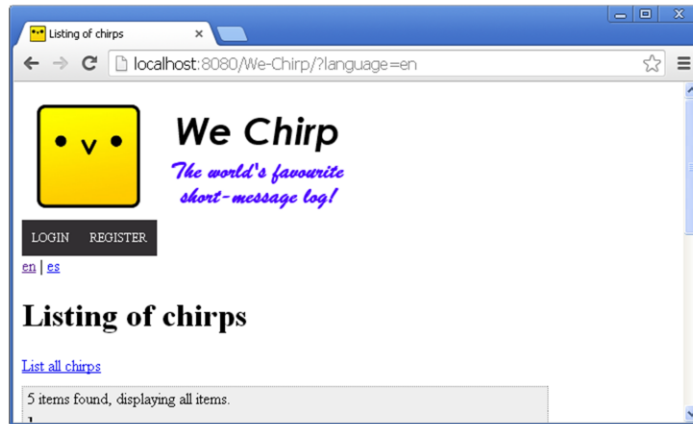


This is our roadmap for today. It won't be too long since the goal's that you can start working on your project as soon as possible. We'll start with a user's tour; then we'll present the UML domain model, and then will comment on the TODOs that we've left open in the project that we provide to you.



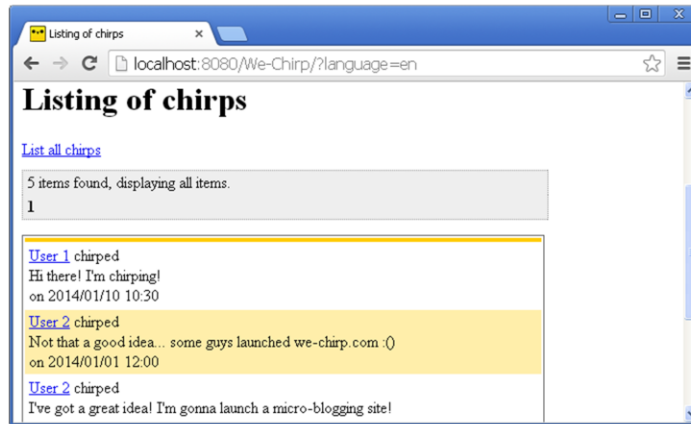
Let's start with the user's tour. In this section, our goal's to present a number of screenshots so that you can have a better understanding of what we expect from this project.

The welcome screen (I)



This is the welcome screen. By now, you should be very familiar with it. It has a logo, a simple menu that just displays “login” and a language bar. Scroll a little down to learn about the differences with other projects.

The welcome screen (II)



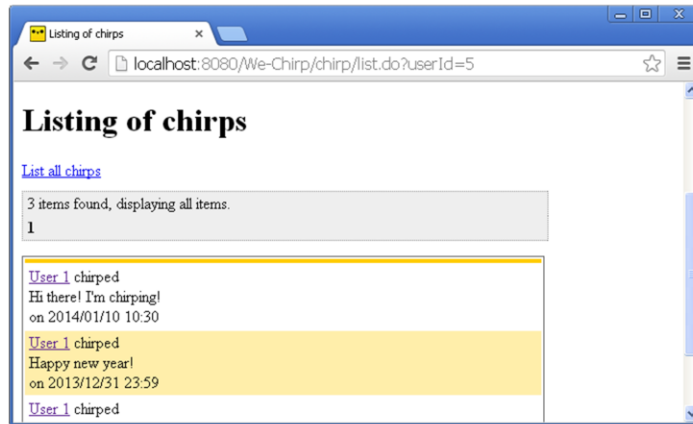
Usually, the welcome screen shows just a welcome message. In this project, it shows the list of chirps that were published in the last thirty one days. The welcome screen allows to scroll through the chirps; it shows the latest ones at the top and then the older ones. A chirp is a message that a user's published; they're never deleted, but they're considered far too old after thirty one days and they're never displayed in this listing.

Focusing on a user's chirps (I)



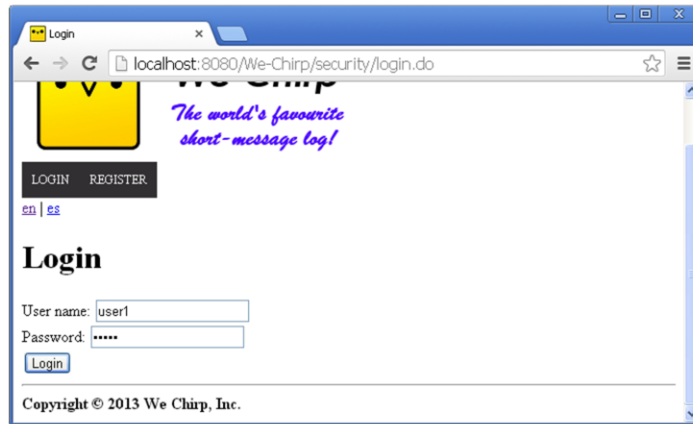
You can focus on a user's chirps by clicking on his or her name. For instance, in this slide the user's clicking on "User 1".

Focusing on a user's chirps (II)



This results in a similar listing in which only the chirps that were published by "User 1" are displayed. Note that the identifier of the user is passed as a URL parameter and that all of the messages that were published by that user are shown in this listing, independently from whether they are older than thirty one days or not.

Logging in as a user



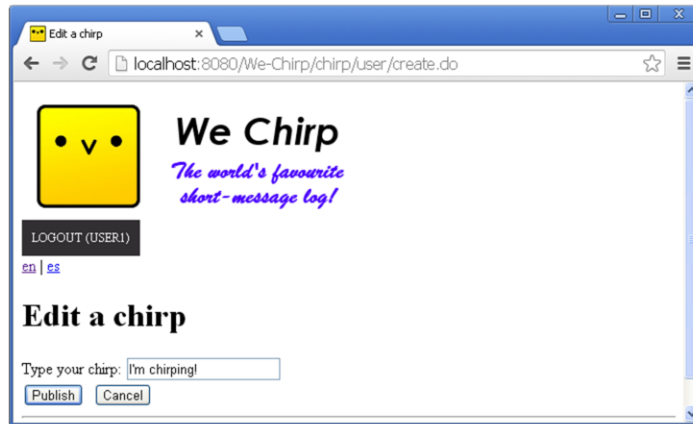
You may log in as a user using two pre-defined user accounts: “user1”/”user1” and “user2”/”user2”.

Chirping a message (I)



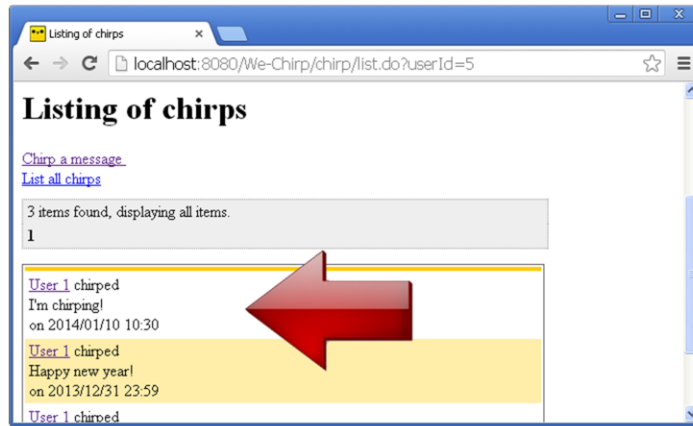
Note that when you log in, the listing of chirps shows an additional link that is entitled "Chirp a message". Click on it.

Chirping a message (II)



This is the user interface to edit a chirp. It's very simple: there's a single text box in which the user can type in a message and a button to publish it or to return to the previous listing. Note that chirps can't be edited once they are published.

Chirping a message (III)



See the new chirp at the top of the listing.

Registering as a user (I)



Note that this application allows users to register themselves. Click on the “Register” option in the menu bar.

Registering as a user (II)

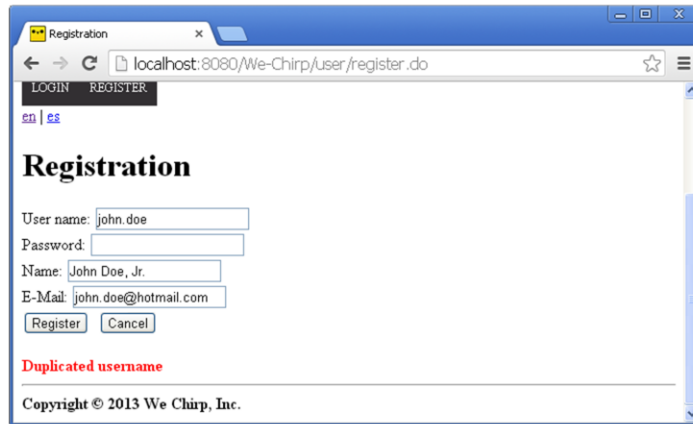
The screenshot shows a web browser window with the title 'Registration'. The address bar displays 'localhost:8080/We-Chirp/user/register.do'. A yellow box highlights the 'short-message log!' link. Below the header, there are 'LOGIN' and 'REGISTER' buttons, and links for 'en' and 'es'. The main heading is 'Registration'. The form contains the following fields and values:

- User name: john.doe
- Password: *****
- Name: John Doe
- E-Mail: john.doe@mail.com

At the bottom of the form are 'Register' and 'Cancel' buttons. The footer text reads 'Copyright © 2013 We Chirp, Inc.'.

This slide shows the form to edit a user's data. He or she can enter a username, a password, his or her name, and an email address.

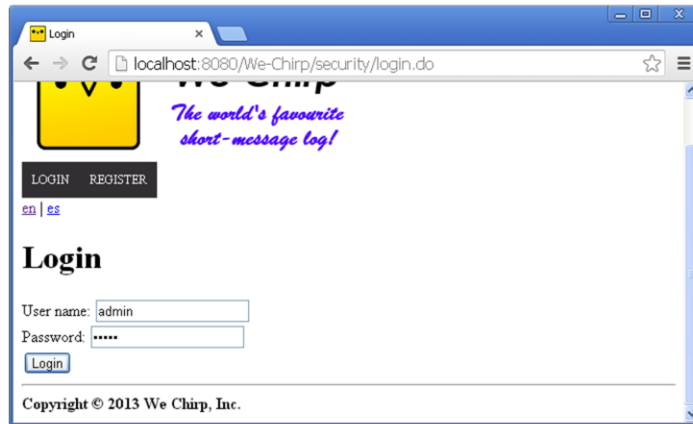
Registering as a user (III)



The screenshot shows a web browser window titled "Registration" with the address bar displaying "localhost:8080/We-Chirp/user/register.do". The page has a navigation bar with "LOGIN" and "REGISTER" links, and a language selector "en | es". The main heading is "Registration". Below it, there are four input fields: "User name:" with the value "john.doe", "Password:", "Name:" with the value "John Doe, Jr.", and "E-Mail:" with the value "john.doe@hotmail.com". There are "Register" and "Cancel" buttons. A red error message "Duplicated username" is displayed below the form. At the bottom, it says "Copyright © 2013 We Chirp, Inc."

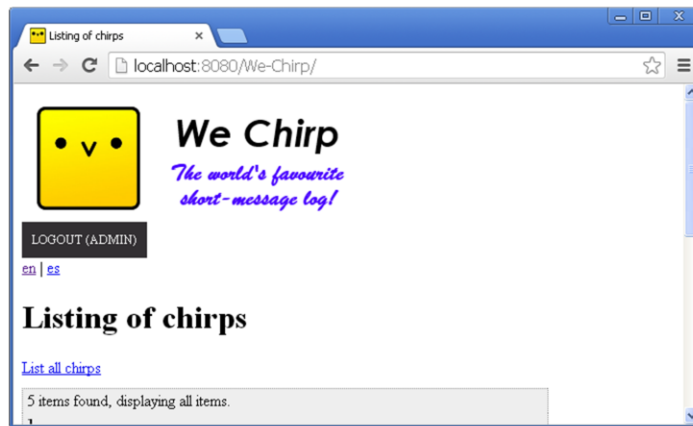
If the user selects a username that corresponds to an already existing user, the application returns an error message like in this slide. Note that it's not a generic error message of the form "Cannot commit this operation", but a customised error message to inform that the username's duplicated.

Logging in as an administrator



You can also log in as an administrator using account “admin”/”admin”.

The administrator's options



In this case, the main menu doesn't offer any options, except for logout. In other words, we won't implement any administrator's functionalities.

You're so inquisitive!

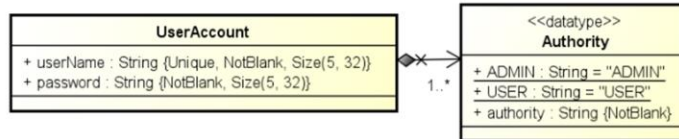


Yes, the conclusion is that “We chirp” is kind of a Tweeter. Chirping’s a synonym for tweeting 😊.



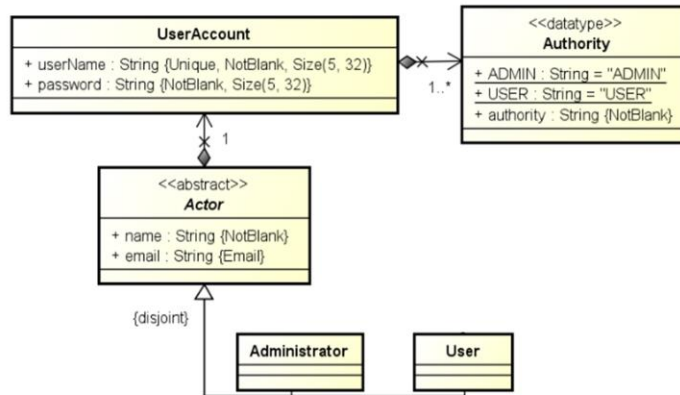
Let's now present the UML domain model.

The model (I)



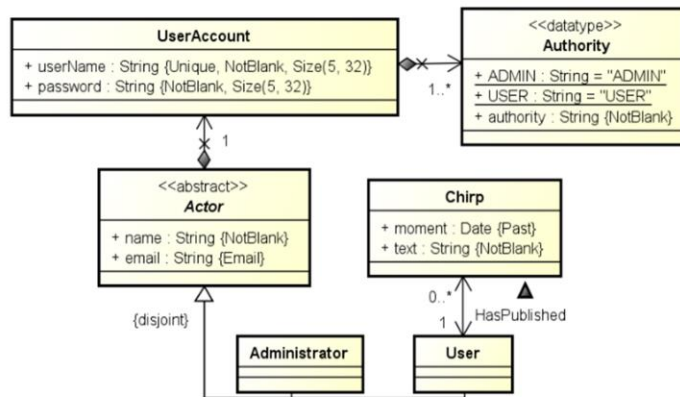
As usual, we'll start with the account-related classes. In this problem, we have only two authorities, namely: administrators and users.

The model (II)



This slide shows the actor hierarchy; as usual there's a class to represent each of the authorities. Note that we've simplified the "Actor" class to include only a name and an email.

The model (III)



This slide shows the model regarding chirps: each user may publish an arbitrary number of chirps and every chirp is published by a single user. It's quite a simple model.



Let's finally take a look at the TODOs on which you have to work so that you can complete the project that we provide.

Converters

- Implement a string-to-authority converter
- Implement an authority-to-string converter
- Implement a string-to-chirp converter
- Implement a chirp-to-string converter
- Implement a string-to-user converter
- Implement a user-to-string converter

This is what you have to do regarding converters. Just create a couple of them and register them.

Repositories

- There are two repositories called `ChirpRepository` and `UserRepository`
- Create the queries that are specified in the corresponding TODO comments

This is what you have to do regarding repositories. There are a couple of them and you have to create a number of JPQL queries that are specified in TODO comments.

Services

- There are two services called ChirpService and UserService
- Implement their methods as described in the corresponding TODO comments

This is what you have to do regarding repositories. There are a couple of them and you have to implement their methods as described in the corresponding TODO comments.

Views

- Create the views to list and edit chirps
- There are some TODO comments to guide you

You must create the views to list and edit chirps. As usual, there are a number of TODO comments that provide a few hints. The view to register a user is provided with the project template. Please, take a look at how we've overridden the default view so that it displays the listing of chirps that were published during the last thirty one days, cf. file `misc/index.jsp`.

Controllers

- There are two public controllers called `ChirpController` and `UserController`
- Complete them using the `TODO` comments that are provided in the code
- There's an additional controller called `ChirpUserController` that you can take as an example.

There are two public controllers called “`ChirpController`” and “`UserController`”. You must complete them using the suggestions that are provided in the corresponding `TODO` comments. There's an additional controller called “`ChirpUserController`” that you can use to learn a couple of techniques that will help you implement the other controllers.

Configuration files

- Complete `converters.xml`
- Complete `icons.xml`
- Complete `tiles.xml`
- Complete `security.xml`

Finally, you'll have to complete a few configuration files, namely: "converters.xml", "tiles.xml", and "security.xml". Please, follow the instructions in the TODO comments.

Ready to start working?



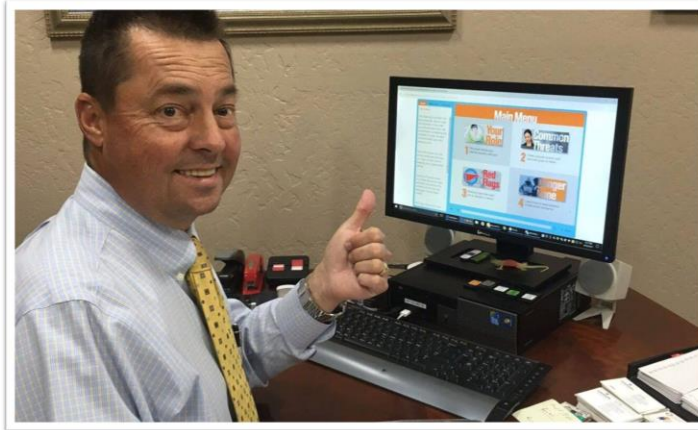
Ready to start working? C'mon!

Take this seriously!



Please, take this problem seriously. We've realised that many students don't work on their problems, and they typically get in trouble. The "We-Chirp" project's quite good to get trained.

Trained for your interview test!



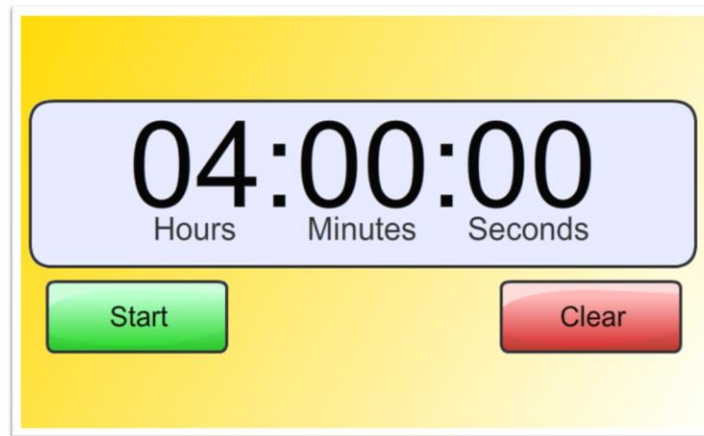
First of all, it's very good to get trained for your interview test. Recall that if you wish to earn a good salary, you must prove that you deserve it. Typically, companies interview their prospective workers and challenge them with a simple project like "We Chirp" before they make a decision on whether to hire them or not.

And for your D&T control check!



But it's also very important to get trained for your D&T control check. Recall that you have to pass an individual control check if you wish to pass the subject; working on the "We Chirp" project will allow you to put almost every theoretical concept to practice.

This is the time you'll be granted



Typically, you'll be granted a maximum of four hours to solve these problems. You should be able to solve this problem in no more than four hours. If you cannot, then you have to work on your weak points and improve. The best way to improve is to repeat every one of the problems that we've suggested in the previous lectures.



Thanks for attending this lecture!