Hello Emanuel,

very straight forward solution. I especially like to point out your approach that the error message you display in case of missing/incorrect input is built up incrementally.

While looking at your ShortAddress class, I noticed that you declared the relevant instance variables as being "protected". It obviously makes sense since you are directly referring to those variables in your FullAddress class and without "protecting" them, they would not be visible at that point.

Following the rule " Use private unless you have a good reason not to" (Oracle 2013), I was wondering if there is a "good" reason in the case of this week's DQ1.

Since your classes both provide getters and setters to access the instance variables you could even declare your instances variables as being "private" and use the relevant methods to access the values.  
But even without getters/setters, I think you could declare the instance variables as being private since the "toString" method is the only "spot" where you access them:  
Your FullAddress method toString could simply call its super class's "toString" (ShortAddress) method and append the new values:

                @Override

                public String toString() {

                               return super.toString() + String.format("%s: %s\n%s: %s\n%s: %s\n%s: %s\n ",

                                                               "House number", houseNumber,

                                                               "Street1", street1Name,

                                                               "Street2", street2Name,

                                                               "City", cityName

                                                               );

                } // end of toString method

What do you think?

With best regards,  
Daniel

References:  
Oracle (2013) 'Controlling Access to Members of a Class'. [Online]. Available from http://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html [Accessed August 8th 2013]

Hi Daniel,

Thank you for looking at my code. Good point about the private vs. protected declaration. I never thought about calling the ShortAddress’s toString() method using super, and simply just adding to it.

Emanuel

Hi John,

I gave your program a try and it worked. The only recommendation I have is to have the results display in a dialog box. I initially tried out your program by just double clicking your application class file, and when I was finished entering data, it just closed, without me seeing the results. I had to run it from the unix terminal to see the short or long address.

Regards,

Emanuel

Hi Jim,

I like the way you set up your GUI. I had a similar GUI, though different very different code. Did you think about making Street 2 optional? Here in the U.S., it’s usually an optional field (e.g., Apartment number), so that’s how I made mine. But I’m not sure how addresses are in the UK.

Emanuel

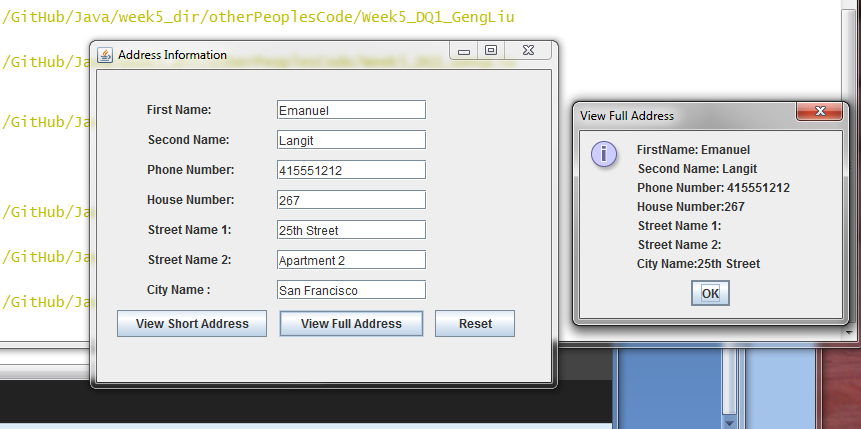
Hi Geng,

I tried your code and had a couple of problems. As you can see from the screenshot, the street name was put into city, and the street name 2 and city did not display. I ran it from the unix prompt, and when I closed the window, I did not get the prompt back, so I had to type ctrl-c. I think this could be solved by adding the following to your main method:

addressUI. setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

Regards,

Emanuel



Hi James,

I tried out your program, and it works. The only suggestion I have is the sizing of the window. When I ran it, I was a little confused about what to enter into each text field because of the way it was arranged:

So I had to manually resize it:

I would give it different size setting in your InitMessage class, by modifying your line:

GUI.setSize(200, 600)

Regards,

Emanuel