```java
/**
 * Function that initializes the add customer form. Contains both lambda expressions.
 * Lambda 1:
 *      parameter 1 = observableValue.
 *      parameter 2 = s.
 *      parameter 3 = t1.
 *      ->
 *      expression = By listener, retrieve division id from value from division combo box selection
 *      and return division id, then used in list single country function.
 * Lambda 2:
 *      parameter 1 = observableValue.
 *      parameter 2 = s.
 *      parameter 3 = t1.
 *      ->
 *      expression = upon action, if no choice is made in division combo box prior, set combo box to list of
 *      divisions of selected country comboBox item.
 * @param url path location.
 * @param resourceBundle source through resources.
 */
@Override
public void initialize(URL url, ResourceBundle resourceBundle) {
    System.out.println("*************************************************************************************");
    System.out.println("*************************************************************************************");
    System.out.println("Initializing AddCustomerController.");
    System.out.println("*************************************************************************************");
    System.out.println("*************************************************************************************");
    System.out.println();

    System.out.println("*************************************************************************************");
    System.out.println();
    System.out.println("REQUIREMENT 2: - 'Customer IDs are auto-generated, and first-level division");
    System.out.println("(i.e., states, provinces) and country data are collected using separate combo boxes.'");
    System.out.println();
    System.out.println("*************************************************************************************");
```

```
    // Lambda 1.
    divisionCombo.valueProperty().addListener(((observableValue, s, t1)
            -> listSingleCountry(customerSQL.divisionIdFromDivisionName(JDBC.connection, divisionCombo.getSelectionModel().getSelectedItem()))));

    // Lambda 2.
        countryCombo.valueProperty().addListener(((observableValue, s, t1) -> {
            if (divisionCombo.getSelectionModel().getSelectedItem() == null) {
                String selectedCountry = countryCombo.getSelectionModel().getSelectedItem();
                listCountryDivisions(selectedCountry);
            }
        }));
}
```

```java
/**
 * Function that deletes a selected appointment once delete button in week tab is clicked.
 * Lambda 3:
 *      parameter 1: selection (button).
 *      ->
 *      expression: conditional reaction based on confirming / deleting appointment.
 *      NOTE: Also in mDeleteApptClicked but hardly any different.
 * @param actionEvent required for action load.
 */
1 usage
public void wDeleteApptClicked(ActionEvent actionEvent) throws SQLException {
    System.out.println("*****************************************************************************");
    System.out.println("Checking for a valid selection in week tab view.");
    System.out.println("*****************************************************************************");

    if(weekTableView.getSelectionModel().getSelectedItem() != null && weekTab.isSelected()) {
        System.out.println("*****************************************************************************
        System.out.println("Selection present/valid. Setting confirmation message for delete confirmation.");
        String confirmation = "This will remove the selected appointment permanently.";
        System.out.println("*****************************************************************************

        ButtonType confirm = new ButtonType( s: "Ok");
        ButtonType cancel = new ButtonType( s: "Cancel");
```