Assignment 7 – Train Tickets with MVC

Goals

- Create a UI that allows the user to specify a train trip they want
- Use MVC techniques to separate layers of program functionality
- Use an explicit intent to launch a new activity that displays the order to the user
- Allow the user to either purchase or edit the order
- On original activity, update display based on their choice to confirm or cancel order (this means you must use **startActivityForResult** with launching the second activity)
- Allow the user to enter and edit Card information on multiple activities

Required naming convention (replace # with the current assignment number)

- **Application Name**
  - A#
- **Company Domain**
  - firstname.lastname.itp341
- **Package Name** (should be automatically generated)
  - itp341.lastname.firstname.a#.app
- **Zip File** (include entire project folder)
  - A#.Lastname.FirstName.zip

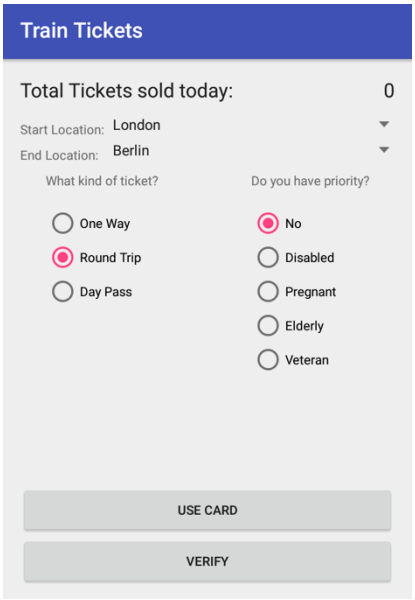
Hint: Passing objects in intents

- Any custom object can be put into an **extra** and passed via an **intent**
- All that is required is that the class be **serializable**, which is a way of representing an object as a series of bytes, which can then be saved
- To make a class **serializable**, the class must implement the serializable interface
  `public class MyAwesome class implements serializable { … }`
- To put an object into the intent, use
  `putExtra(…)`
- To retrieve an object from an intent
  `getSerializableExtra(…)`

Requirements

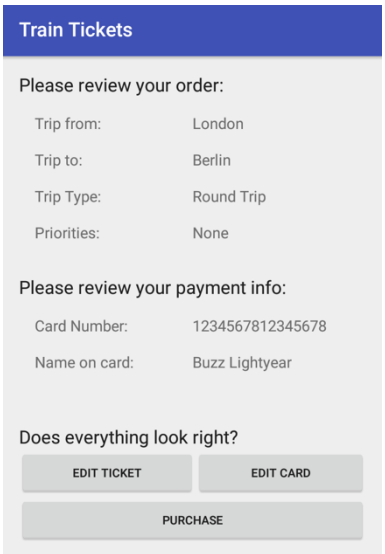
- Create new Android Application Project
- Follow default prompt, but make sure to choose **Empty Activity**
- Model
  - Create a model class for a Ticket (**Ticket**) and a Card (**Card**)
  - These classes should be POJO and include the standards elements:

        - Methods
            - Constructor
            - toString
            - getters and setters
        - Instance variables for all the following (naming up to you)
            - Ticket
                - Starting location
                - Ending location
                - Trip type
                - Priorities
            - Card
                - Number
                - Name
                - Security code (optional)
- UI View – **activity_main.xml**
    - Follow the rules in the Android Design Guide. This includes height of widgets, spacing, padding, etc. http://developer.android.com/design/style/metrics-grids.html
    - Create a similar layout to the one below



- Activity – **MainActivity**
    - Best Practices Reminder
        - Be sure to create class members that will refer to the widget you want to access later
        - This means you should only call **findViewById** in the **onCreate** method
    - Configure listeners for buttons

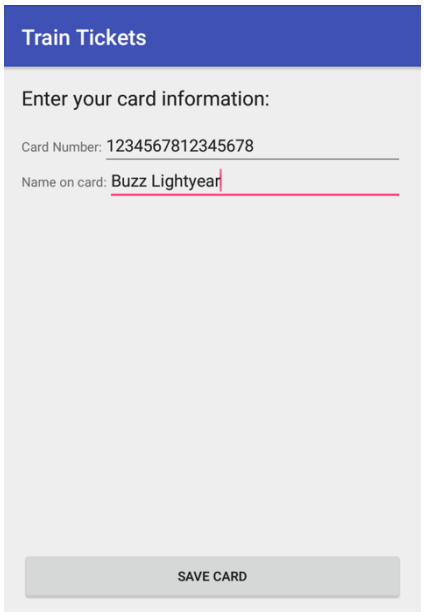
        - See below
- Listener – **MainActivity**
    - When the user clicks the **Use Card** button:
        - Launch an intent to go to a **CardActivity**.
        - The Intent should be executed in such a way that MainActivity can take back the data (a **Card**) and use it later.
    - When the user clicks the **Verify** button:
        - The specifications for the order must be packaged into a **Ticket** object
        - The Intent should be executed in such a way that **MainActivity** will know how the status of how **ViewOrderActivity** ended
        - Pass the Ticket and any existing Card objects through the intent, not simply the values.
            - Hint: How can we modify the Ticket class to allow it to be passed in an intent?
- UI View – **activity_view_orders.xml**
    - Follow the rules in the Android Design Guide. This includes height of widgets, spacing, padding, etc. http://developer.android.com/design/style/metrics-grids.html
    - Create a similar layout to the one below



    - Layout must present all the user's order option in a visually-pleasing manner
- Activity – **ViewOrderActivity**
    - Best Practices Reminder
        - Be sure to create class members that will refer to the widget you want to access later
        - This means you should only call **findViewById** in the **onCreate** method
    - Configure listeners for buttons

        - See below
- Listener – **ViewOrderActivity**
    - Create a listener for each button
    - When the user clicks the **Purchase** button:
        - This means the order is correct so let **MainActivity** know that the user confirmed the order
        - Be sure to return the most up to date **Card** when the activity finishes.
        - Call **finish()** to end the current activity
    - When the use clicks **Make Edit:**
        - This means the order is incorrect so let **MainActivity** know that the user canceled the order
        - Be sure to return the most up to date **Card** when the activity finishes.
        - Call **finish()** to end the current activity
    - When the user clicks **Edit Card**:
        - This means the **Card** displayed needs to be changed so launch an Intent to move to the **CardActivity**. When you return from the activity, save the value of the **Card** and update the UI.
- Activity – **MainActivity**
    - After returning from **ViewOrderActivity**, you should do one of the following
        - If user clicked **Edit Card**
            - Make sure to store the updated card for future use.
        - If user clicked **Purchase**
            - Clear all the fields in the form
            - Display a **Toast** thanking the user for their order
            - Update the "Tickets" counter at the top of the Activity
        - If user clicked **Make Edit**
            - Make sure all the previous specifications from the user order are in the form
            - Display a **Toast** asking them make any changes needed
- UI – View – activity_card.xml
    - Follow the rules in the Android Design Guide. This includes height of widgets, spacing, padding, etc.
      http://developer.android.com/design/style/metrics-grids.html

- Create a similar Layout to the one below



- Security Code is optional.

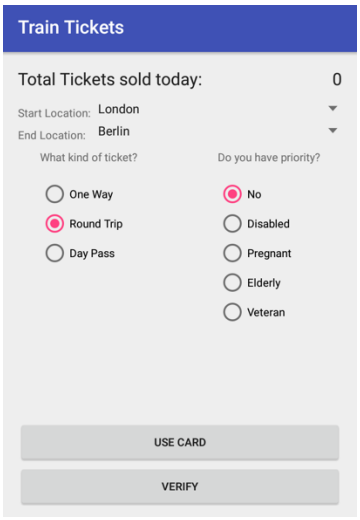
- Activity – **CardActivity**
  - Best Practices Reminder
    - Be sure to create class members that will refer to the widget you want to access later
    - This means you should only call **findViewById** in the **onCreate** method
  - Configure listeners for buttons
    - See below
- Listener – **CardActivity**
  - When the user clicks **Save Card**, create a **Card** object and pass it back to the parent activity (returning immediately using the **finish()** method)
  - Hint: Since both **MainActivity** and **ViewOrderActivity** can launch **CardActivity**, it might be better that all constants for **Cards** are stored in **CardActivity**

Extra Credit

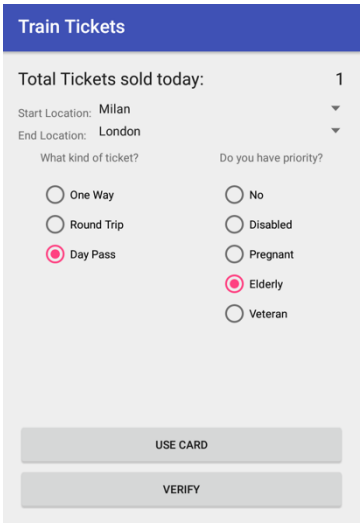
- (10 Points) Use SharedPreferences to retain the last modified **Card** information even when restarting the app.

Sample Output

Main Activity

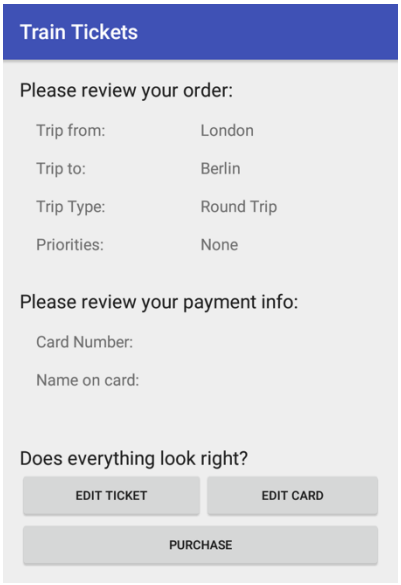


Before Verifying Order

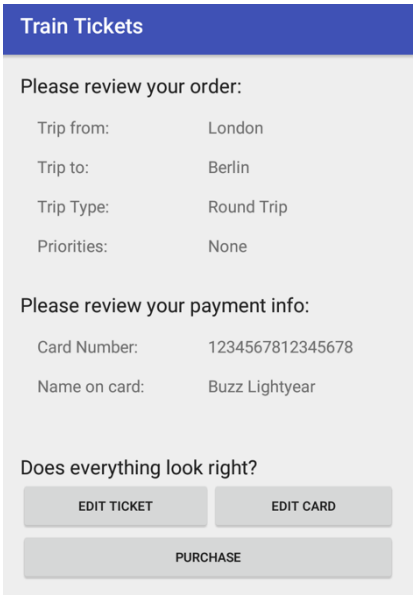


After Verifying Order

ViewOrderActivity



Before entering Card info



After entering Card info

Deliverables

1. A compressed file containing your app. Follow the guidelines for full credit.
   Here are the instructions for submission
   a) Navigate to your project folder.
   b) Include the entire folder in a zip file
   c) Upload zip file to Blackboard site for our course

Grading

Item	Points
Ticket and Card POJO	5
Main activity responds to specification and updates after Verify	10
Main activity starts View Orders and Card activities	10
Ticket and Card POJO is passed between activities	10
View Orders layout meets specifications	5
View Orders displays the latest information	5
Card Activity meets specifications	5
Total	**50**