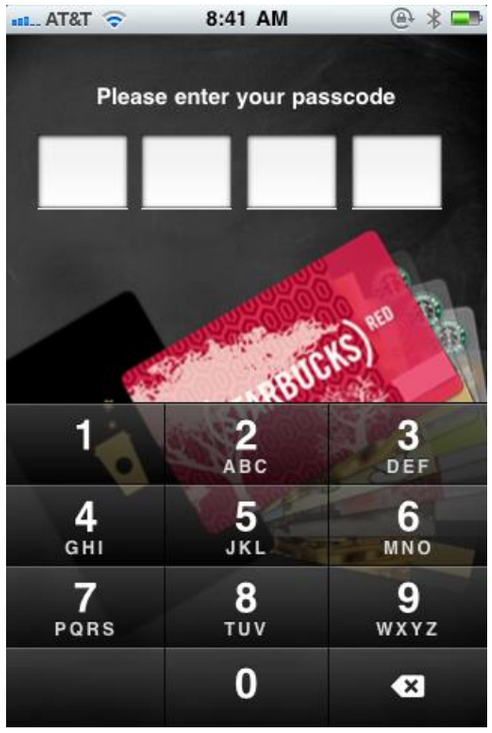
Starbucks

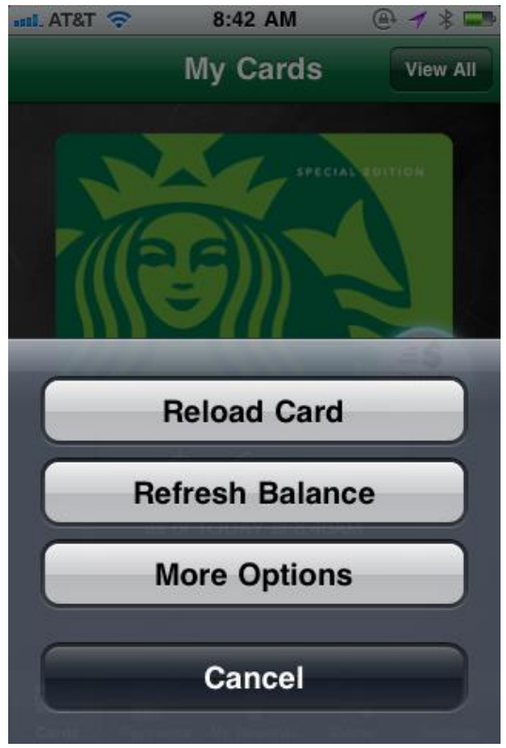
Mobile App Simulator Project Requirements
2018 Version



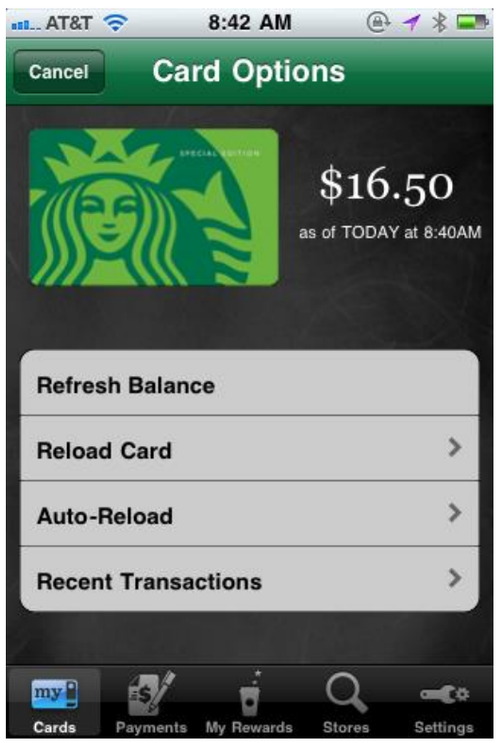
Pin Screen

My Cards - Main

My Cards - Pay

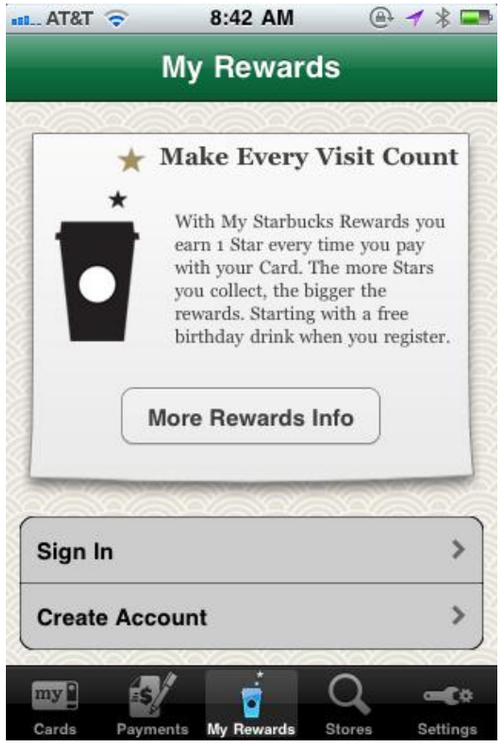


My Cards - Options

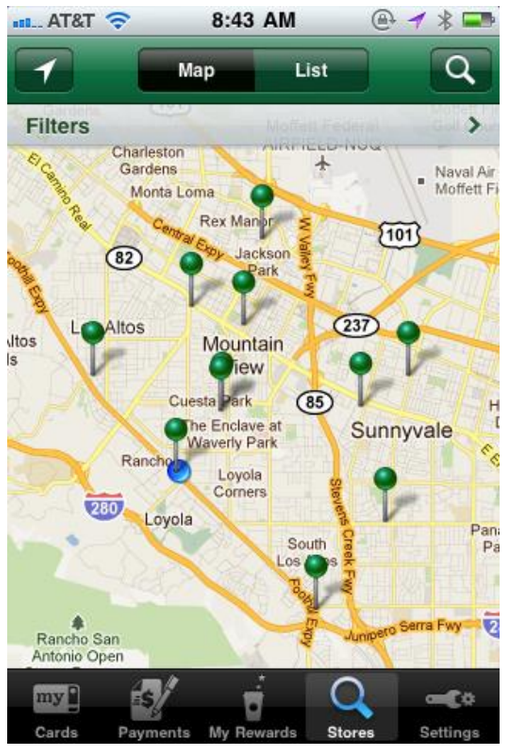


My Cards
More Options

Payment Setup



Rewards Setup



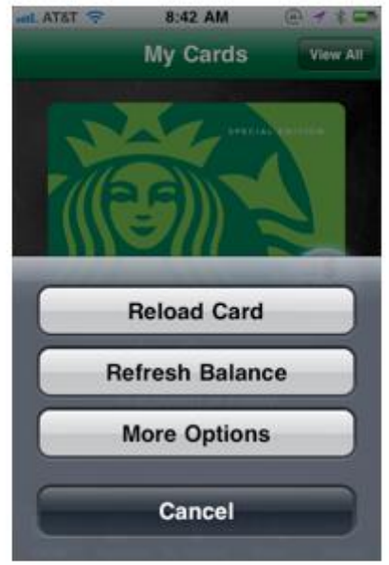
Find Starbucks

Solution should implement an “**App Controller**” class which should contain a “display()” method to display the current ***Screen Name*** as well as well as call the “***display()***” method of the current Screen each time the Screen changes

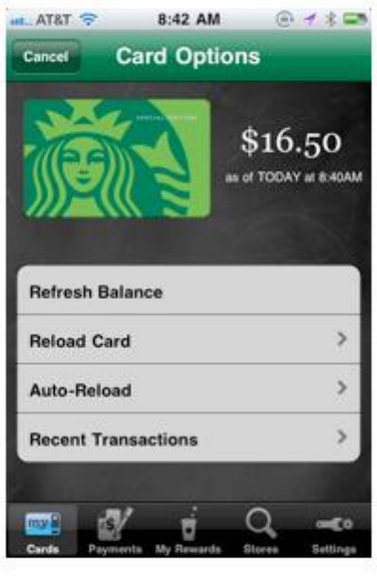
Pin Screen

My Cards - Main

My Cards - Pay



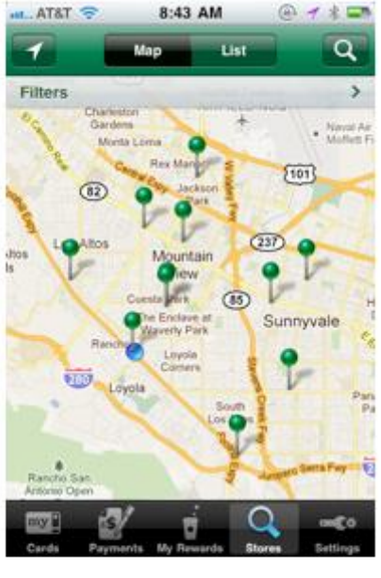
My Cards - Options



My Cards
More Options

Payment Setup

Rewards Setup



Find Starbucks

Additionally, ***App Controller*** implements the ***IApp*** Interface, which includes the following operations:

1. **Orientation Operations** (landscape, portrait)
2. **Screen Display** (to render screen contents)
3. **Touch Input** (to send touch events to Screen components)
4. **Execute Command** (to initiate one of five possible commands on menu bar)
5. **Screen Navigation** (next / previous screen)
6. **Screen Name** (to get the Class name of current Screen)
7. **Screen Contents** (to get the on screen content of current Screen)

```
4   public interface IApp
5   {
6
7       void landscape() ;           // switch to landscape view
8       void portrait() ;            // switch to portrait view
9       void touch(int x, int y) ;   // send touch event to current screen
10      void display() ;             // display contents of current screen
11      void execute( String c ) ;   // trigger a nav bar menu item
12      void prev() ;                // navigate to previous screen
13      void next() ;                // navigate to next screen
14      String screen() ;            // get name of the current screen
15      String screenContents() ;    // get contents of current screen
16
17  }
```

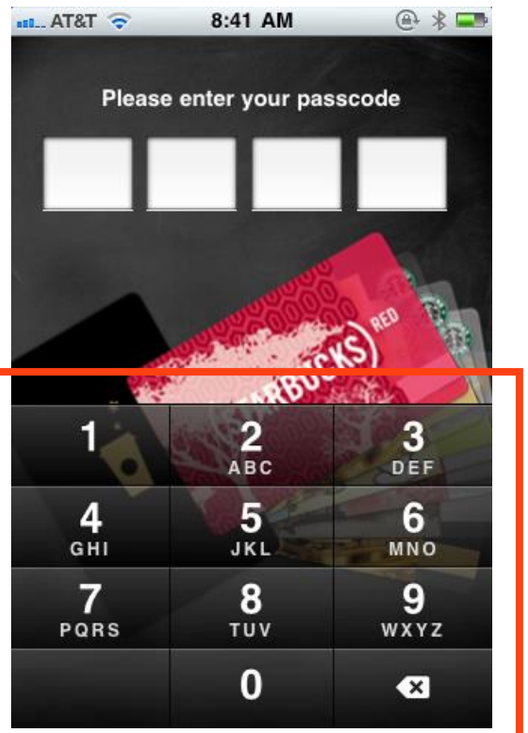
All Screens should implement a common interface which includes "***display()***" and "***touch(x,y)***" methods. The "***touch(x,y)***" method accepts the coordinates below. Screens can contain components (such as the "keypad") which have their own relative coordinates. Touch events can be mapped from the Screen coordinates into the widget's relative coordinates. For example, given the following layout of the PinScreen, a "***touch(1,5)***" would map into a "***keyPress(1,1)***" on the KeyPad.

```
5   public interface IScreen
6   {
7
8       void touch(int x, int y) ;              // send touch events to screen
9       String display() ;                      // displays screen components
10      String name() ;                         // returns name of screen
11      void next() ;                           // navigate to next screen
12      void prev() ;                           // navigate to previous screen
13      void setNext(IScreen s, String n ) ;    // set next screen with action name
14      void setPrev(IScreen s, String n ) ;    // set previous screen with action name
15
16  }
```

(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

PinScreen Coordinates (x,y)

Pin Screen



(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

(1,1) = **1**	(2,1) = **2**	(3,1) = **3**
(1,2) = **4**	(2,2) = **5**	(3,2) = **6**
(1,3) = **7**	(2,3) = **8**	(3,3) = **9**
(1,4) = _	(2,4) = **0**	(3,4) = **X**

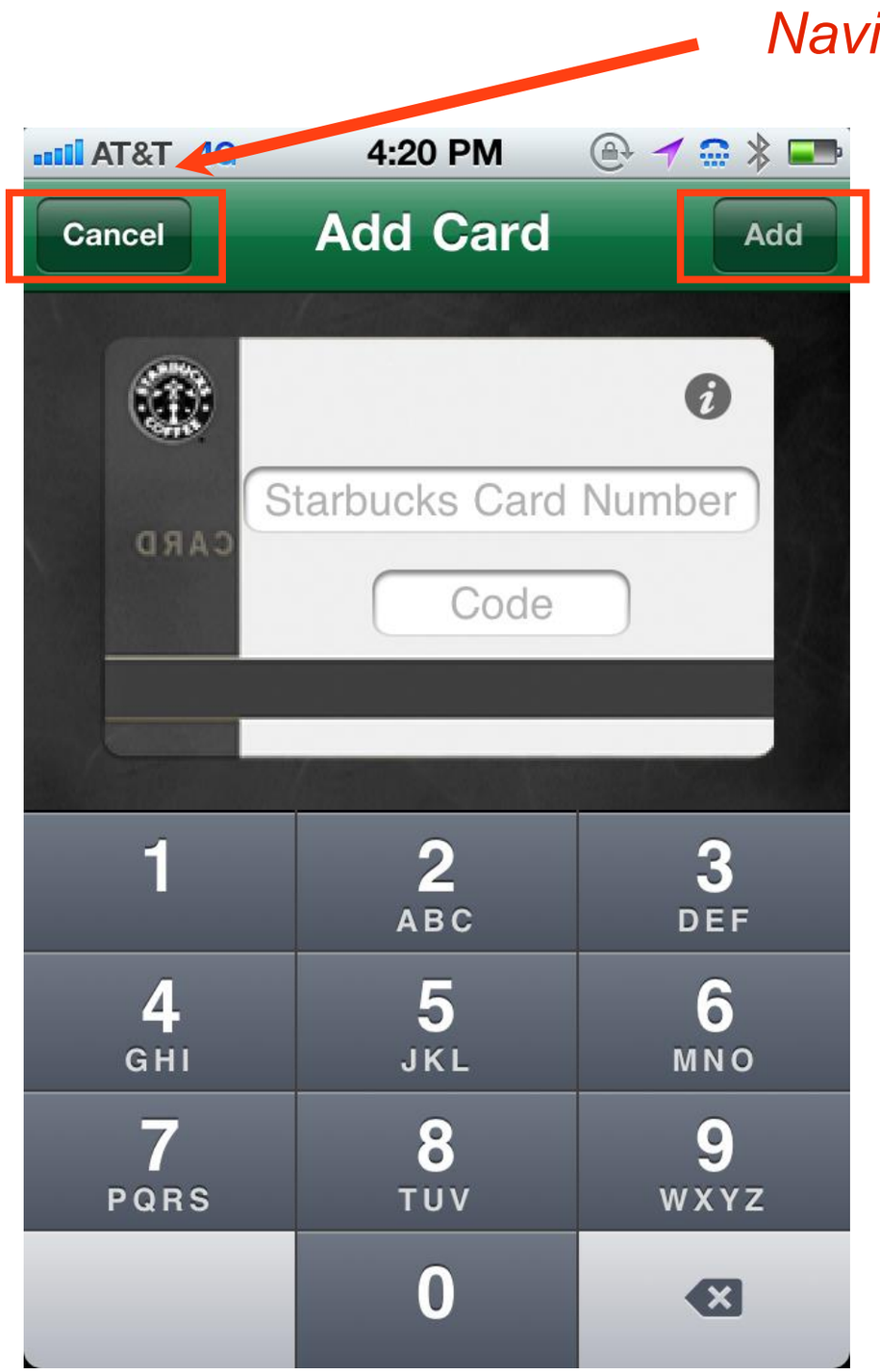
Coordinates: (x,y) = Key

KeyPad
inside PinScreen

Sample Touch Event Mapping:

== >Touch (1,5)
==> KeyPress(1,1)
==> Key Number 1

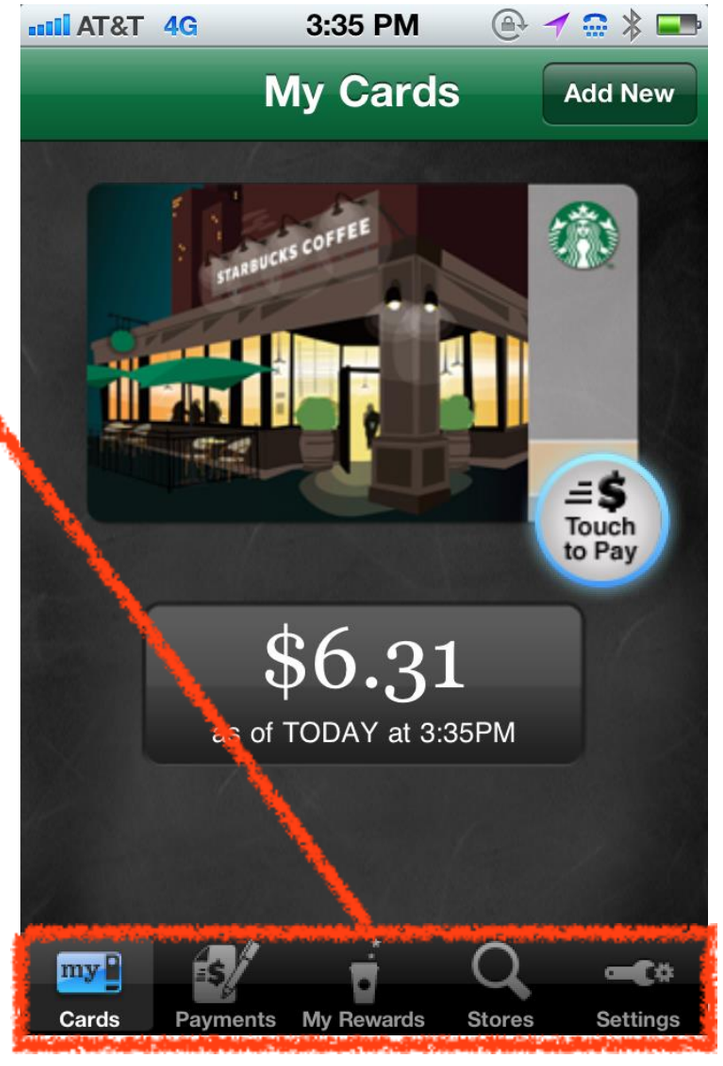
App Controller should implement "***Previous***" and "***Next***" Actions on the ***Top Left*** and ***Top Right*** of the Screen — which would be mapped to certain operations in a particular screen (if any at all). For example, the ***Add Card Screen maps "Previous" to "Cancel"*** and ***"Next" to "Add"***.




```
4   public interface IApp
5   {
6
7       void landscape() ;          // switch to landscape view
8       void portrait() ;           // switch to portrait view
9       void touch(int x, int y) ;  // send touch event to current screen
10      void display() ;            // display contents of current screen
11      void execute( String c ) ;  // trigger a nav bar menu item
12      void prev() ;               // navigate to previous screen
13      void next() ;               // navigate to next screen
14      String screen() ;           // get name of the current screen
15      String screenContents() ;   // get contents of current screen
16
17  }
```

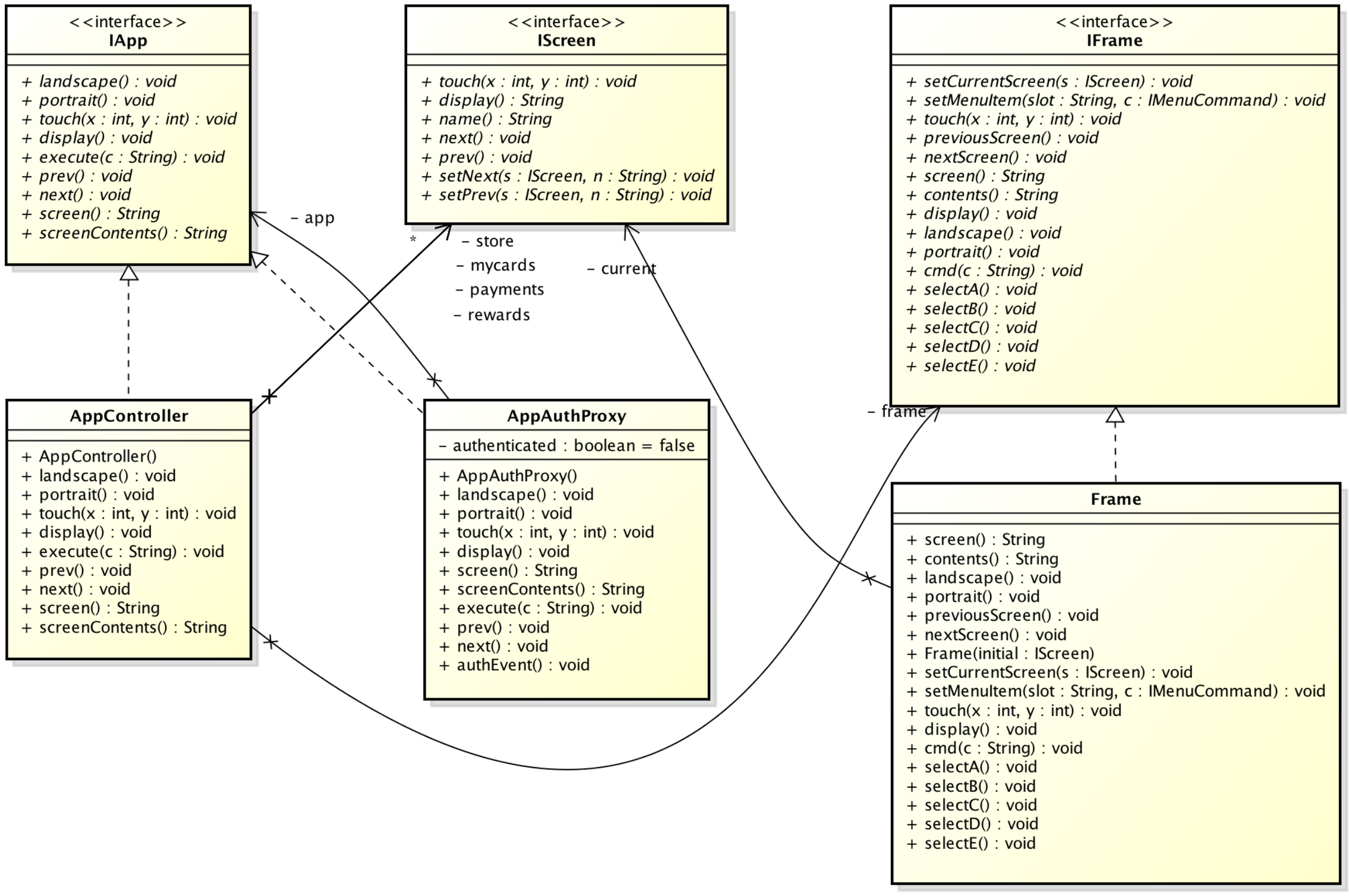
App Controller also must implement a ***Menu Bar*** (typically position at the bottom of an iPhone Screen) via a ***Frame*** *for managing Screen flows*.

The ***IFrame*** Interface should include ability invoke any of these ***menu items*** to represent each of the five menu options an application may have.




```
3   public interface IFrame
4   {
5
6       void setCurrentScreen( IScreen s ) ;
7       void setMenuItem( String slot, IMenuCommand c ) ;
8       void touch(int x, int y) ;
9       void previousScreen() ;
10      void nextScreen() ;
11      String screen() ;
12      String contents() ;
13      void display() ;
14      void landscape() ;
15      void portrait() ;
16      void cmd( String c ) ;
17      void selectA() ;
18      void selectB() ;
19      void selectC() ;
20      void selectD() ;
21      void selectE() ;
22
23  }
```

UML Diagram of *Interfaces* discussed so far:

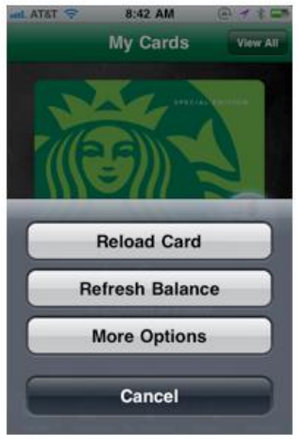


Screen Flows

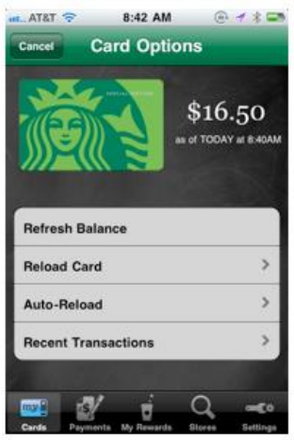
Pin Screen

My Cards - Main

My Cards - Pay

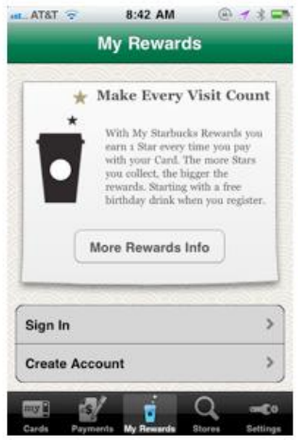


My Cards - Options

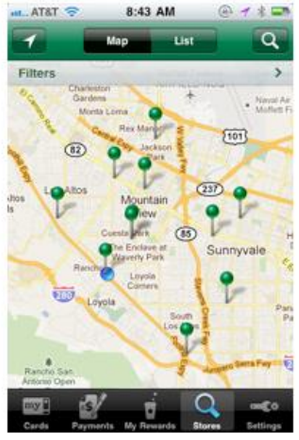


My Cards
More Options

Payment Setup



Rewards Setup



Find Starbucks

Screen Touch Coordinates

(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

Implement the following Screen Flows and Touch Behaviors:

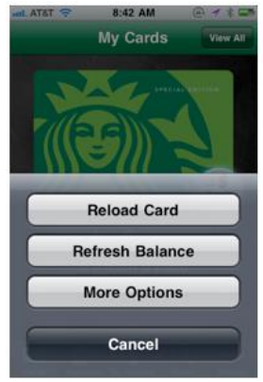
1. After a "**Successful**" Pin Validation, the "**My Cards – Main**" Screen appears
2. A **touch(3,3)** on the "**My Cards – Main**" screen switches to the "**My Cards – Pay**" Screen
3. A **touch(2,4)** on the "**My Cards – Main**" screen switches to the "**My Cards – Options**" Screen
4. A **touch(1,7), touch(2,7) or touch(3,7)** on the "**My Cards – Options**" Screen switches to the "**My Cards – More Options**" screen
5. A **touch(3,3)** on the "**My Cards – Pay**" screen switches to "**My Cards – Main**" Screen.

Display Requirements

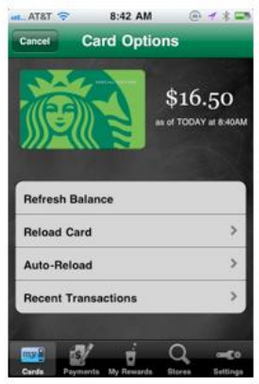
Pin Screen

My Cards - Main

My Cards - Pay

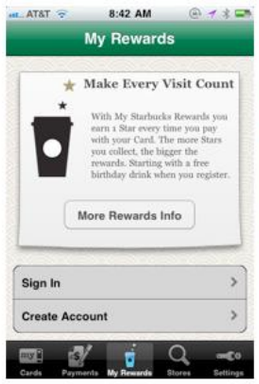


My Cards - Options

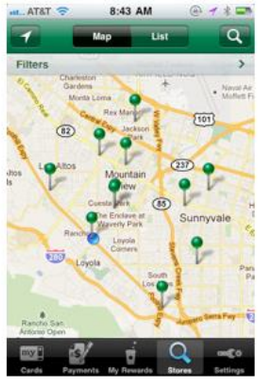


My Cards
More Options

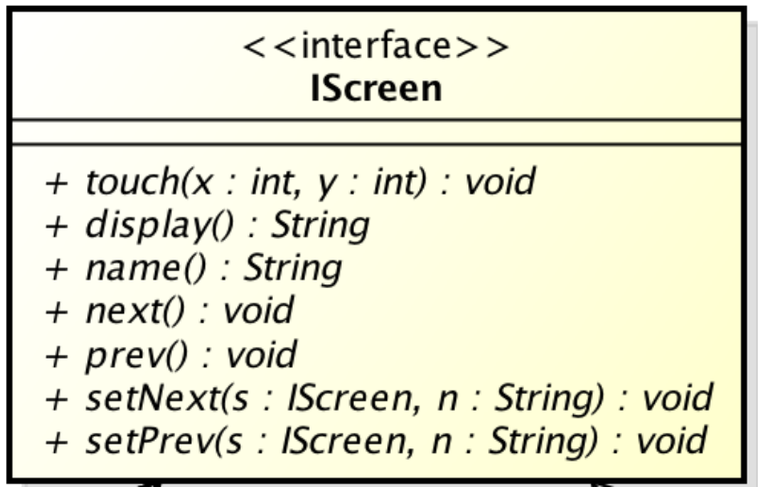
Payment Setup



Rewards Setup

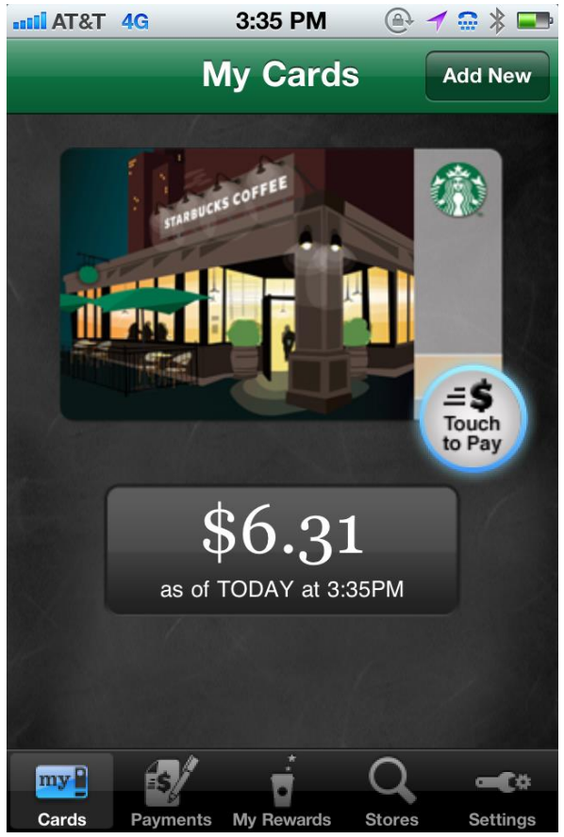


Find Starbucks

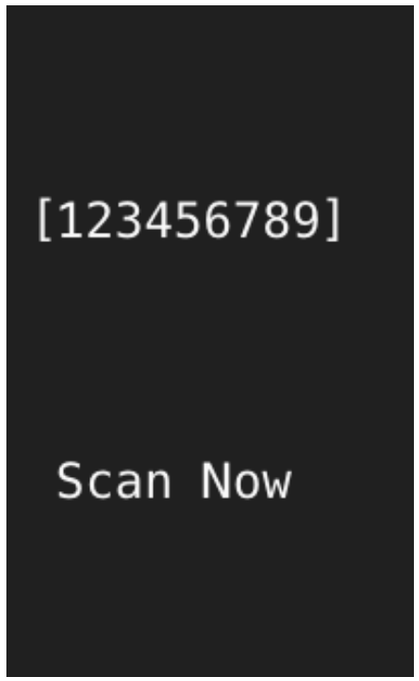


Implement the following Display Output for each screen:

1. Calling "**display()**" on the "**My Cards – Main**" Screen should **Show the balance of the current active card**. (**Example**: **$16.50**)
2. Calling "**display()**" on the "**My Cards – Pay**" Screen should print-out the words "**Scan Now**" along with the **Card ID** of the **active card**.
3. Calling "**display()**" on the "**My Cards – Options**" Screen should print-out the words "**Reload, Refresh Balance, or More Options"**.
4. Calling "**display()**" on the "**My Cards – More Options**" Screen should print-out the words "**Refresh, Reload or View Recent Transactions**"
5. Calling "**display()**" on the "**Payment Setup**" Screen should print-out the words "**Enable Payments?**"
6. Calling "**display()**" on the "**Rewards Setup**" Screen should print-out the words "**Make Every Visit Count**".
7. Calling "**display()**" on the "**Find Starbucks**" Screen should print-out "**Google Map of Local Starbucks**"



The “***My Cards - Main***” Screen initially starts out an “empty” card (i.e.Card ID and code of all zeros and a balance of $0.00). The Screen display**, for example, should show the current Card Balance.**



The “***My Cards - Pay***” Screen should display the **9-digits Card ID** around brackets.

With 3 blank Lines,

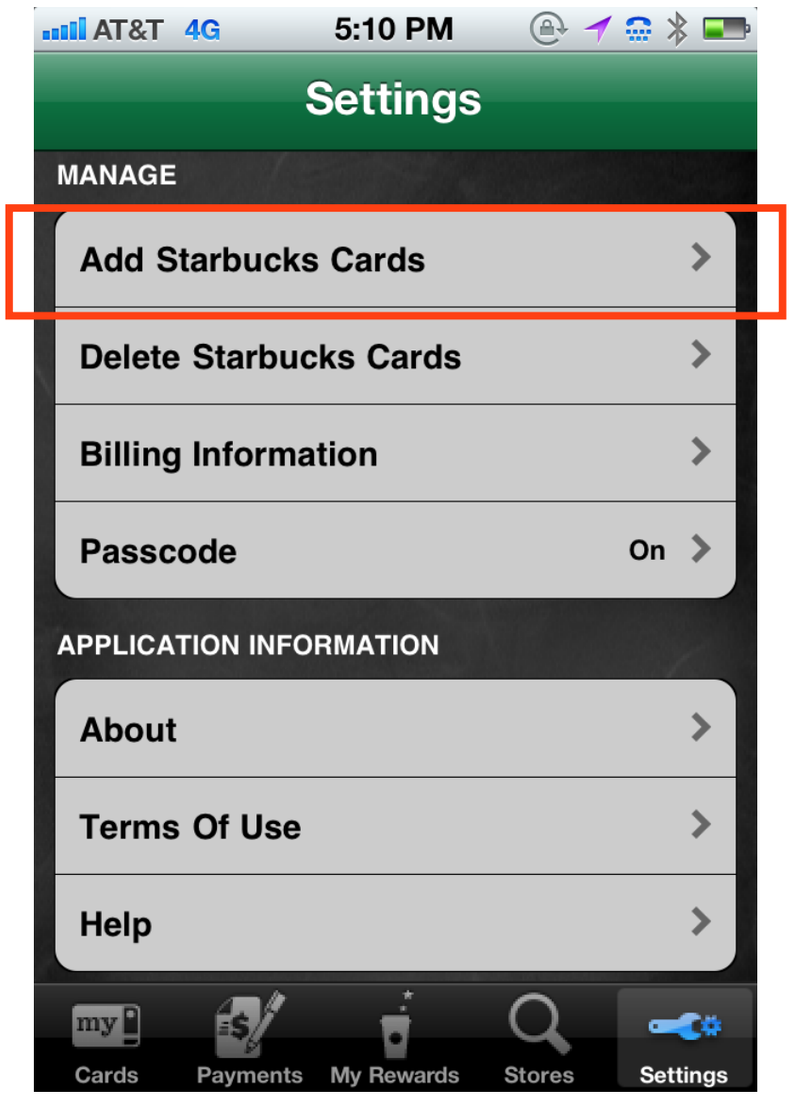
And the text “***Scan Now***”

Settings Screen

To add a Card, a user would touch the "***Settings***" menu and then the "***Add Starbucks Cards***" option. This option has the following touch coordinates: **(1,1), (2,1) or (3,1**).

Design a **Detailed Sequence Diagram** for this workflow, which should result in creating a new Card object making it the new "Default" card for payment.

Additionally, the "***display()***" method for ***Add Card Screen*** should print "***Enter a New Card***"; likewise, the "***display()***" method for ***Settings Screen*** should print "***Manage Card, Billing, Passcode | Show About & Terms.***"



Screen Touch Coordinates

(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

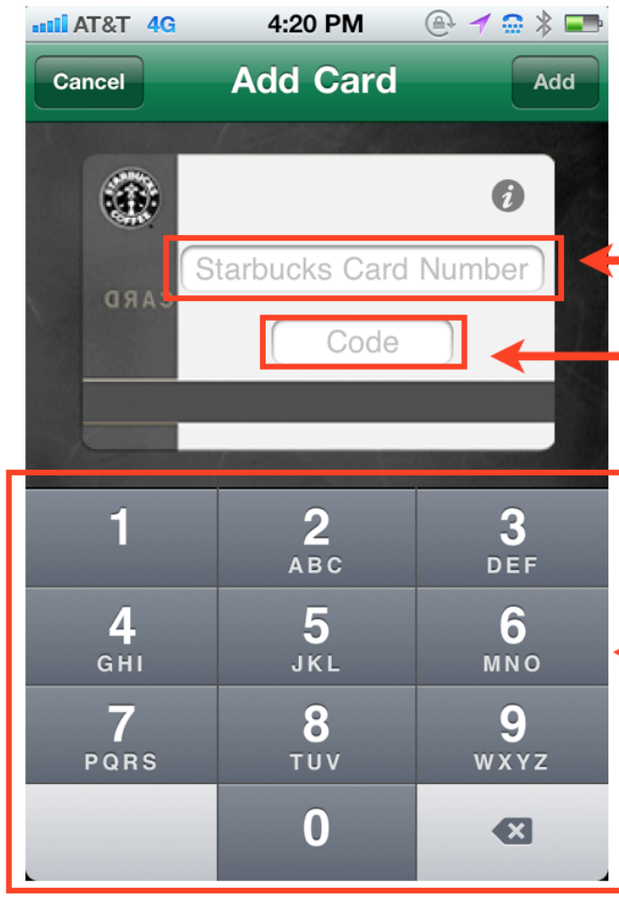
NOTE: Add all new cards with a default initial balance of $20.00

Add Card Screen - New Card Entry

The "***Add Card Screen***" should reuse the **KeyPad** widget. In the "**Add Card Screen**", ***9 digits can be entered for the Card ID*** and ***3 digits can be entered for the Card Code***.

A **touch()** with coordinates of **(1,2), (2,2) or** (**3,2)** should set the **Card ID entry as the current focus.** Likewise, **touch (2,3)** will **switch the focus to the Card Code**. With the focus of the cursor, new digits "touched" should be **appended** to the current "**Card ID**" or "**Card Code**" numbers via the ***KeyPad***.

All New Cards will be added with a default balance of **$20.00** (and will replace any previous Cards entered). That is, the Simulator only manages one card and not a list of Cards. ***Only cards with 9 digits card numbers and 3 digits codes are allowed to be added. If this validation fails, clear out both the Card Number Entry Box and the Code Entry Box and wait for the user to reenter.***

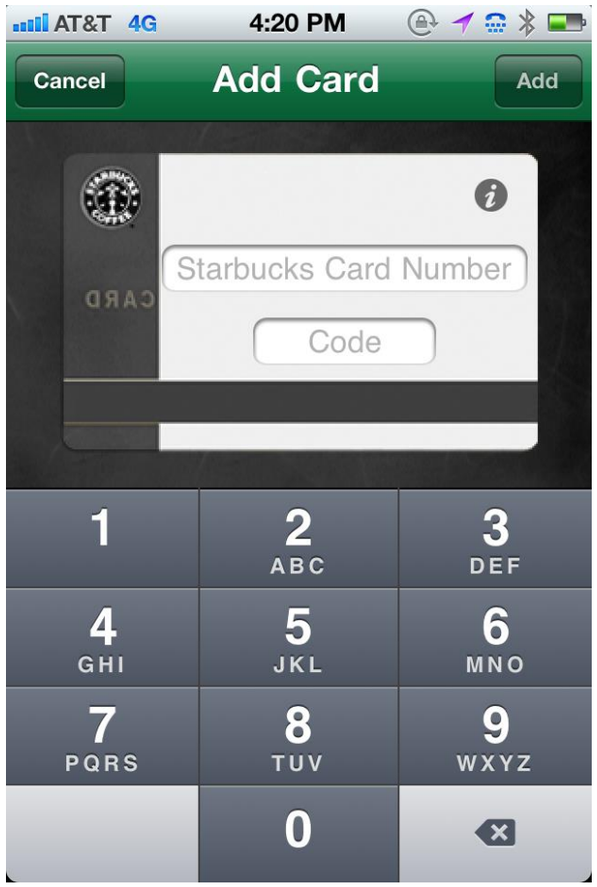


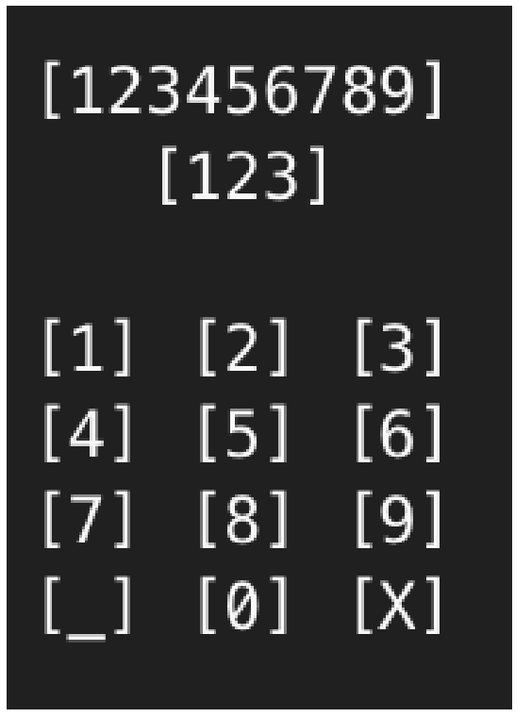
(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

Add Card Screen - Display Format

As Digits are being entered, the "***Add Card***" Screen should Display the following text lines (with **line breaks**) as follows:

Line #1 - The **9 Digits Card ID** around Brackets. I.E. **[123456789]**
Line #2 - The **3 Digits Card Code** around Brackets. I.E. **[123]**
Line #3 - Blank Line
Lines #4 - #7 - **Key Pad Digits**





9 Digits Card ID

3 Digits Card Code

Blank Line

Key Pad Digits

Add Card Screen - Navigation

The "***Next Nav***" on the Add Card Screen maps to "***Add New Card Request***" with the digits entered and returns to "***My Cards - Main***" showing the new card ***balance***.

The "***Prev Nav***" maps to "***Cancel Card Entry***" and returns to the ***Previous Screen (i.e. Settings Screen)***.

Cancel Card Entry & Go Back to Settings Screen

Add "Valid" Card & Display New Card Balance

Making Payments

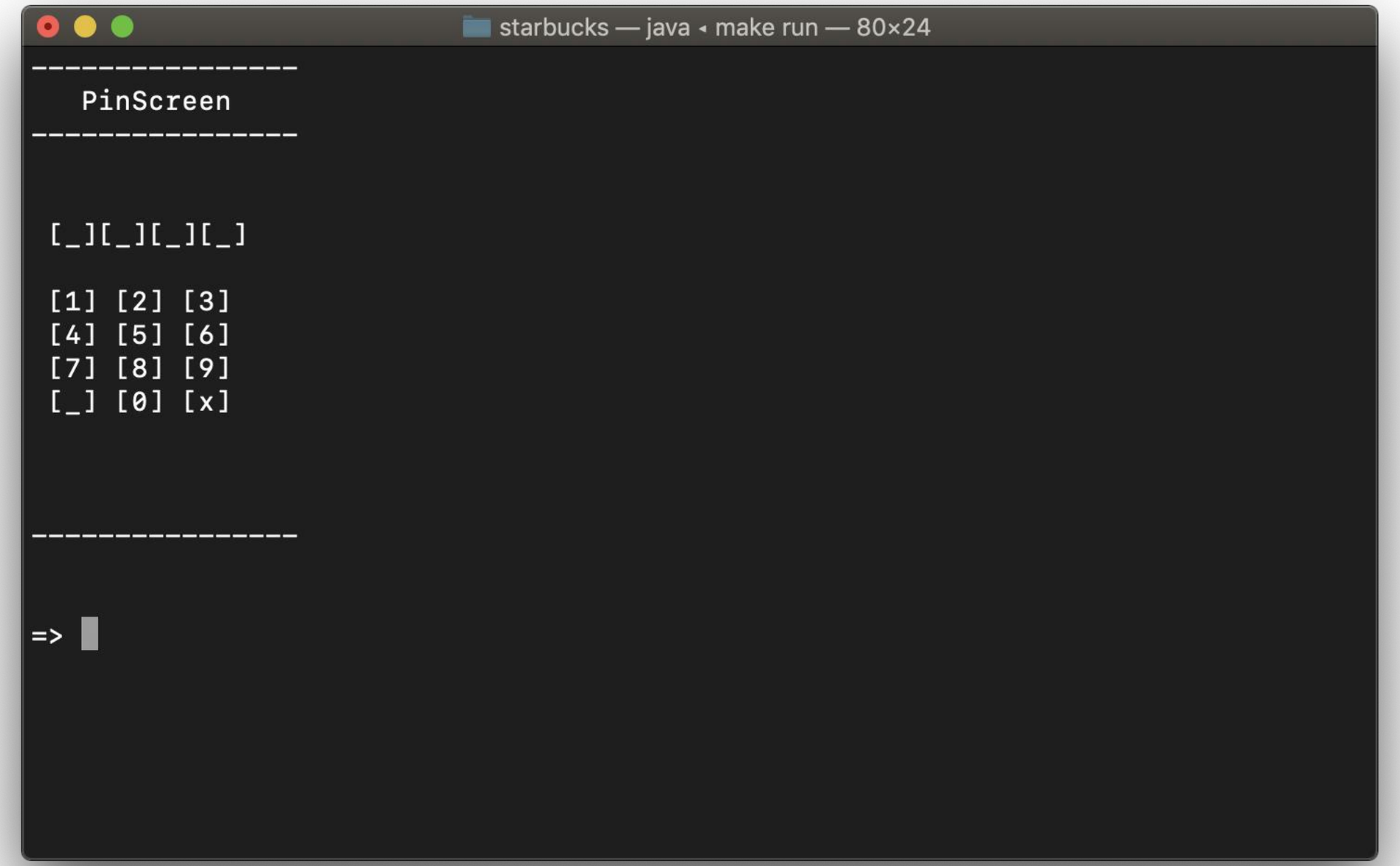
For **Payments**, “Hard Code” a **charge of $1.50 for each transaction** made with the current card. If the Card Balance is below ***$1.50*** do not allow the charge. (*i.e. no error message is required, just don’t deduct from the card balance*)

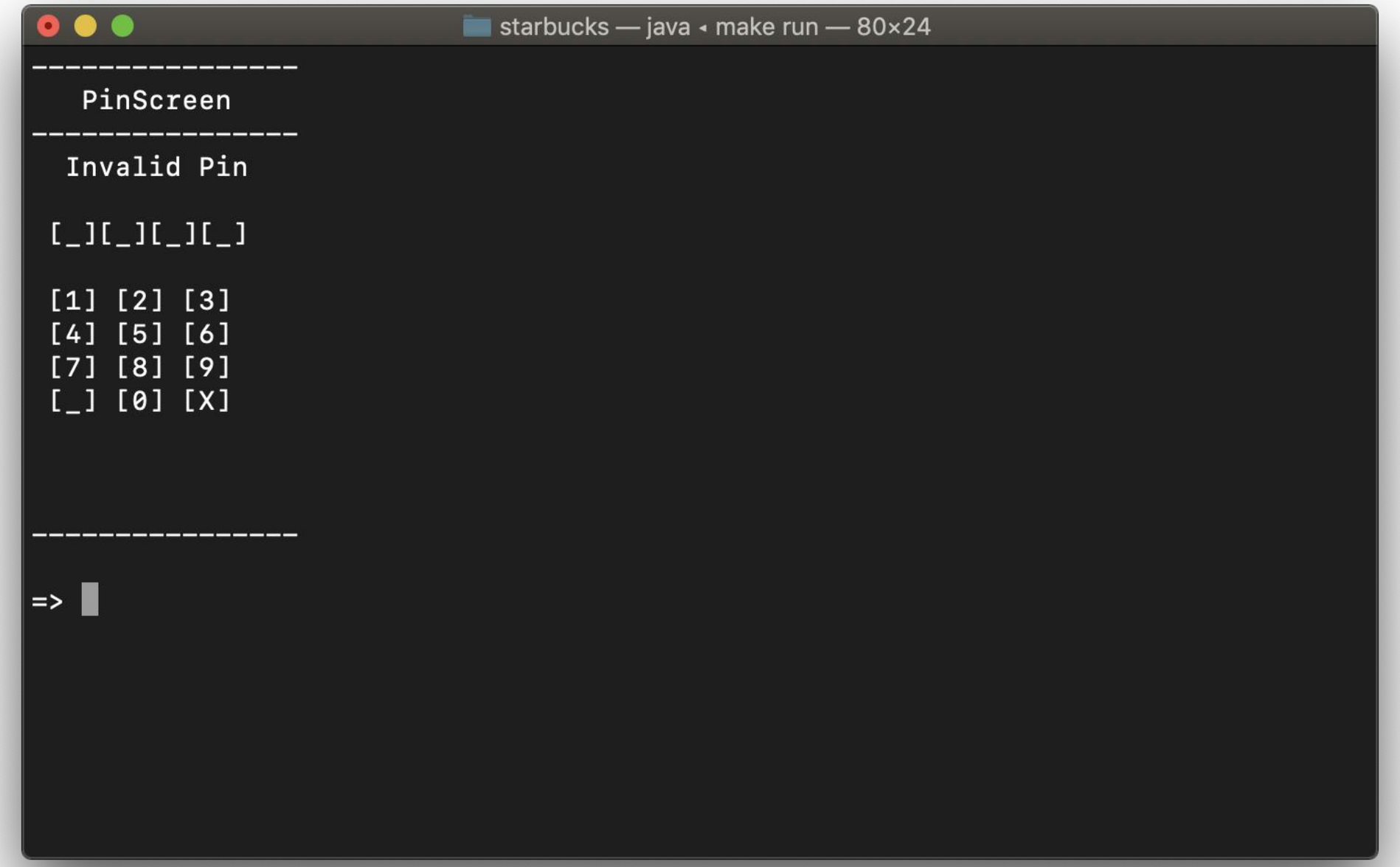
The following touch(x,y) coordinates should trigger the payment:

touch (2, 2), touch (3, 2)

(1,1)	(2,1)	(3,1)
(1,2)	(2,2)	(3,2)
(1,3)	(2,3)	(3,3)
(1,4)	(2,4)	(3,4)
(1,5)	(2,5)	(3,5)
(1,6)	(2,6)	(3,6)
(1,7)	(2,7)	(3,7)
(1,8)	(2,8)	(3,8)

SCREENSHOTS

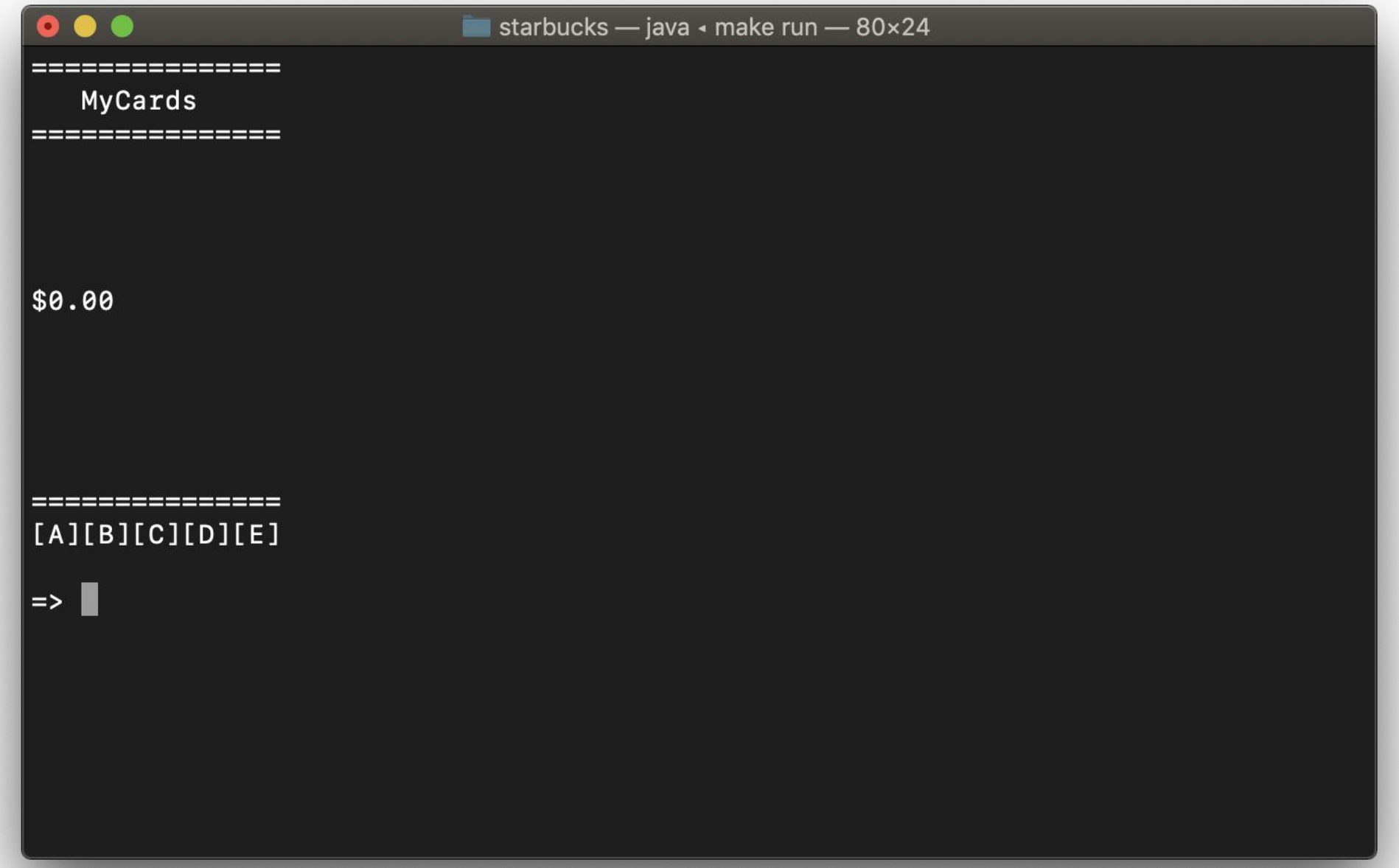

starbucks — java ◂ make run — 80×24
PinScreen
[_][_][_][_]
[1] [2] [3]
[4] [5] [6]
[7] [8] [9]
[_] [0] [x]
=>

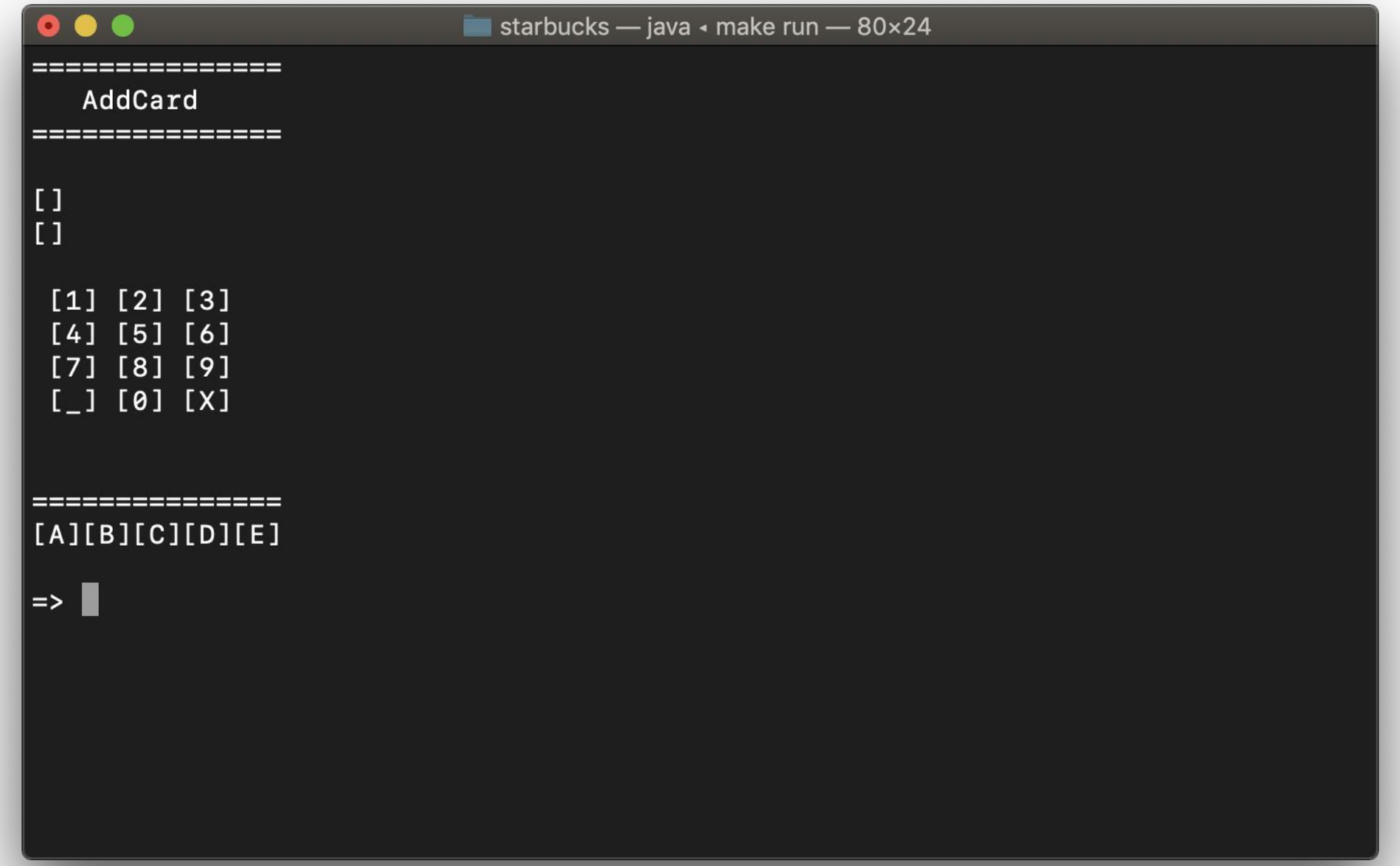

starbucks — java ◂ make run — 80×24

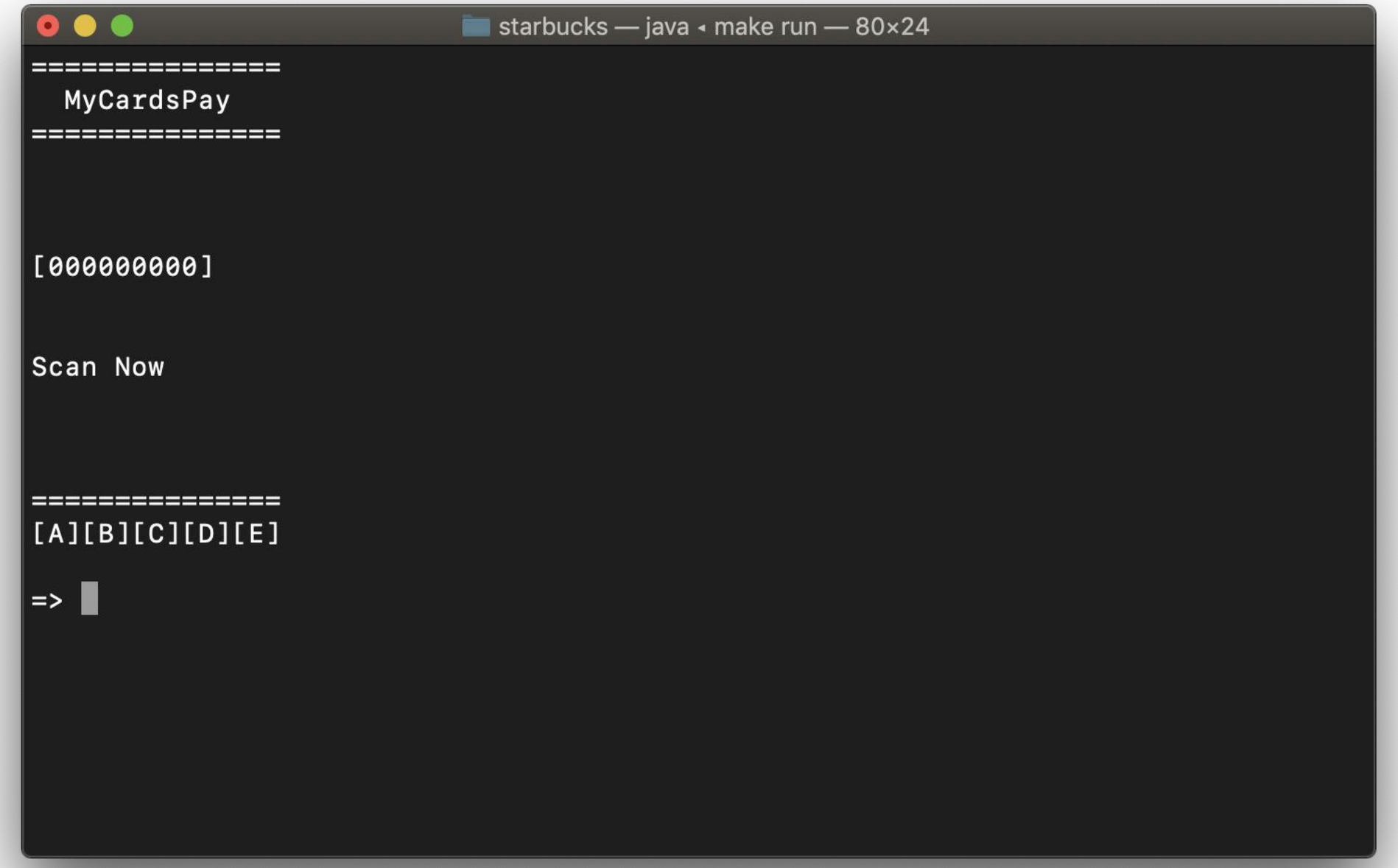
PinScreen

Invalid Pin
[_][_][_][_]
[1] [2] [3]
[4] [5] [6]
[7] [8] [9]
[_] [0] [X]

=>


starbucks — java ◂ make run — 80×24
==========
MyCards
==========
$0.00
==========
[A][B][C][D][E]
=>


starbucks — java ◂ make run — 80×24
==========
AddCard
==========
[]
[]
[1] [2] [3]
[4] [5] [6]
[7] [8] [9]
[_] [0] [X]
==========
[A][B][C][D][E]
=>


starbucks — java ◂ make run — 80×24
==========
MyCardsPay
==========
[000000000]
Scan Now
==========
[A][B][C][D][E]
=>