Card Game Project: High/Low and Blackjack

Card Game Android Project

Brief: to create an Android app to allow the user to play 2 card games: the High Low Game, and a simplified version of Blackjack.

Both games are fairly straightforward.

The High Low Card Game's purpose is to run the logic for the Java classes already defined in the Java part of the course, and display the winner of the game on a new Activity screen.

Brief (cont’d)

The Blackjack game allows the user to play against a ‘Dealer’ (i.e. computer), who plays by using a random generator to decide on the actions to play.

High Low Game Rules:

- Dealer deals two random cards to 2 players in turn
- Hand values are calculated
- The player with the highest hand wins, and the result, together with that player’s cards, are displayed on the Result Activity screen.

Blackjack Rules:

- Normally several players on a table
- But in this version, one Player plays against the Dealer
- Dealer and Player are dealt two cards each
- Player calculates hand value as the sum of the ranks of the cards in hand.
- Ranks equal face value apart from Aces which count for 11 here (in real Blackjack, they can also convert to a 1 if an 11 would cause the Player to Bust (a ‘soft’ Ace)
- Player acts first. Can either Stand (i.e. stop taking further cards, which ends their turn) or Hit (take another card).

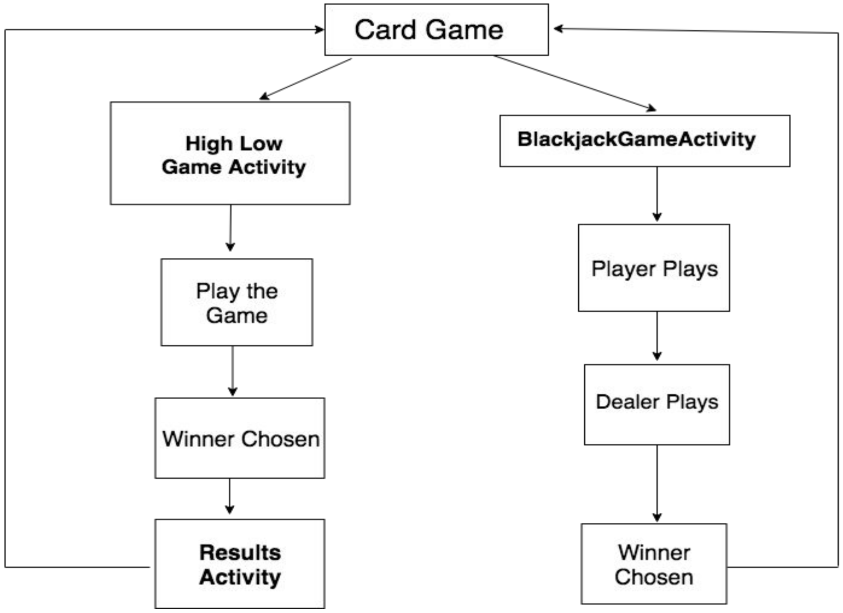
Blackjack Rules (cont’d):

- If the Player’s hand value exceeds 21, the Player busts and the Dealer wins the hand.
- If the Player Stands without going bust, the Dealer then repeats the process of Hit/Stand until they either Stand or Bust
- Once the Dealer’s turn is concluded, if neither player is bust then the hand values are compared to decide the winner
- However if either player has 21 after the first round of 2 cards that player is said to have ‘Blackjack’.
- If both players bust then the dealer wins. If both players have Blackjack, or their hand values tie, then the outcome is a ‘Push’ (draw).

Blackjack Rules (cont’d):

Players win by not busting and having a total higher than the dealer, or not busting and having the dealer bust, or getting a blackjack without the dealer getting a blackjack.

Card Game Site Map



Card Game

High Low Game Activity

Play the Game

Winner Chosen

Results Activity

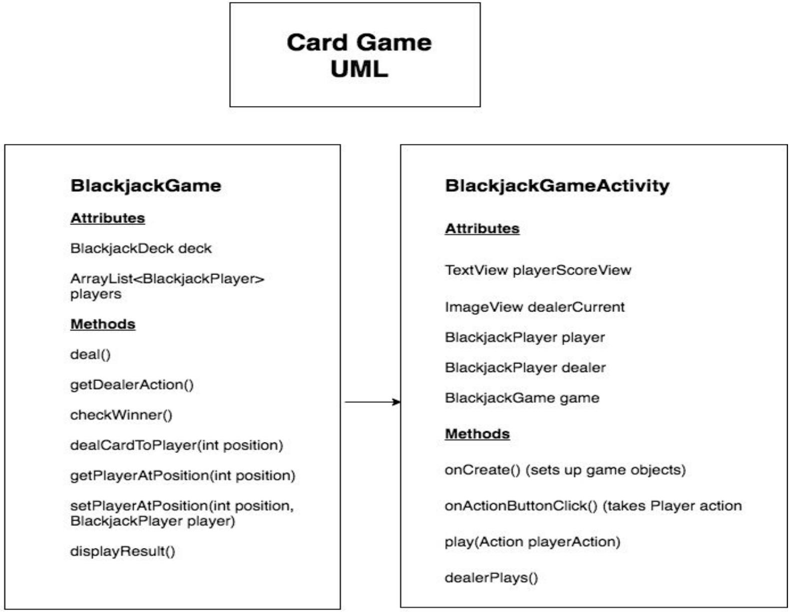
BlackjackGameActivity

Player Plays

Dealer Plays

Winner Chosen

Card Game UML Diagram



Card Game UML

BlackjackGame

Attributes

BlackjackDeck deck

ArrayList<BlackjackPlayer> players

Methods

deal()

getDealerAction()

checkWinner()

dealCardToPlayer(int position)

getPlayerAtPosition(int position)

setPlayerAtPosition(int position, BlackjackPlayer player)

displayResult()

BlackjackGameActivity

Attributes

TextView playerScoreView

ImageView dealerCurrent

BlackjackPlayer player

BlackjackPlayer dealer

BlackjackGame game

Methods

onCreate() (sets up game objects)

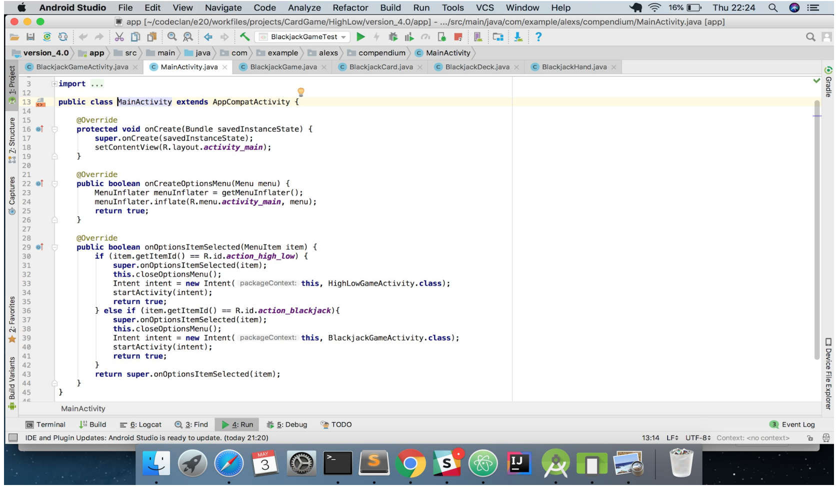
onActionButtonClick() (takes Player action

play(Action playerAction)

dealerPlays()

Code Highlights:

- The Main Activity Screen uses a Menu object to allow users to choose between the High Low Game or Blackjack



Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
16% Thu 22:24
app [~/codeclan/e20/workfiles/projects/CardGame/HighLow/version_4.0/app] - .../src/main/java/com/example/alexs/compendium/MainActivity.java [app]
BlackjackGameTest
version_4.0 app src main java com example alexs compendium MainActivity
BlackjackGameActivity.java MainActivity.java BlackjackGame.java BlackjackCard.java BlackjackDeck.java BlackjackHand.java
1: Project
7: Structure
Captures
2: Favorites
Build Variants
Gradle
Device File Explorer

```java
import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.activity_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId() == R.id.action_high_low) {
            super.onOptionsItemSelected(item);
            this.closeOptionsMenu();
            Intent intent = new Intent( packageContext: this, HighLowGameActivity.class);
            startActivity(intent);
            return true;
        } else if (item.getItemId() == R.id.action_blackjack){
            super.onOptionsItemSelected(item);
            this.closeOptionsMenu();
            Intent intent = new Intent( packageContext: this, BlackjackGameActivity.class);
            startActivity(intent);
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

MainActivity
Terminal Build 6: Logcat 3: Find 4: Run 5: Debug TODO Event Log

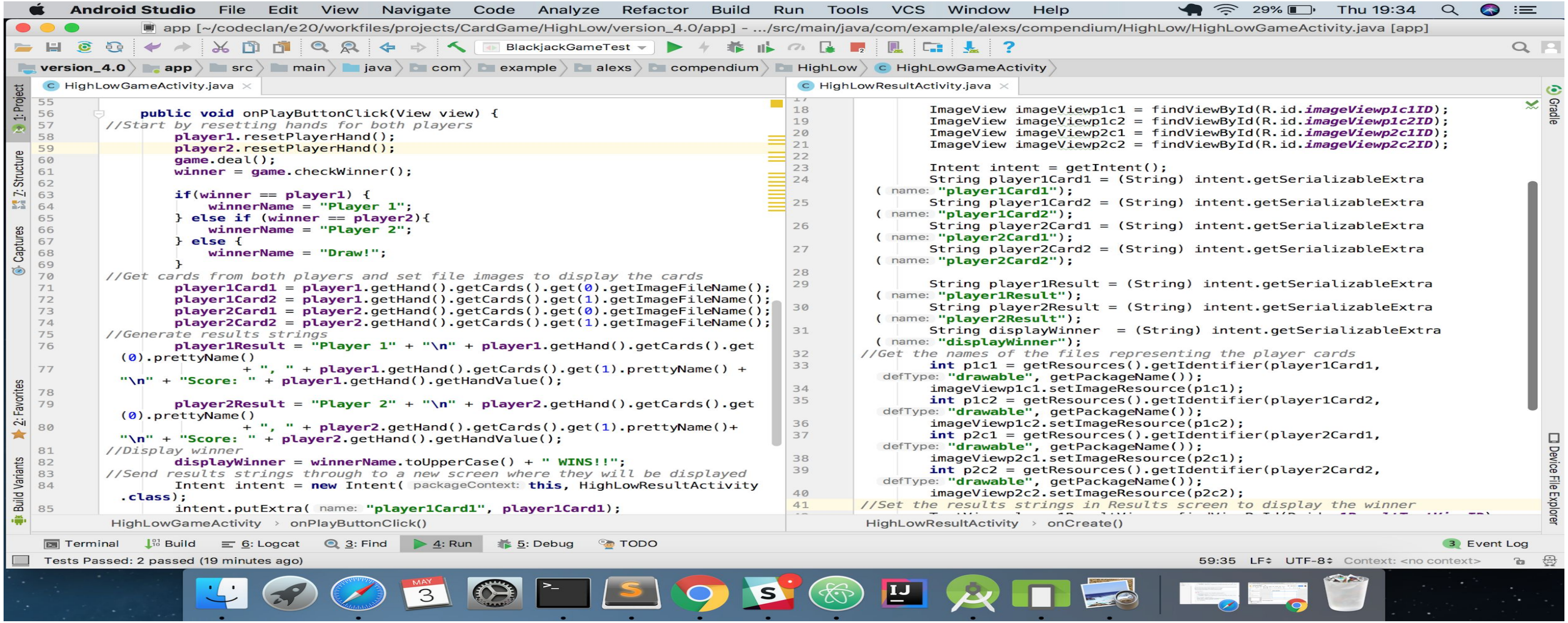
IDE and Plugin Updates: Android Studio is ready to update. (today 21:20)

Code Highlights (Cont’d)

High Low Game:

- Game is run by a Java class called HighLowGameActivity
- Sets up the Hands/Players/Game objects, deals two random cards to both Players, gets the .png card image file names and the name of the winning player
- Sends this information through to another Activity screen, HighLowResultsActivity, using an Intent object. HighLowResultsActivity sets the file card images and displays the results.

High Low Game Activity Screens: Java Classes


Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
29% Thu 19:34
BlackjackGameTest
version_4.0 › app › src › main › java › com › example › alexs › compendium › HighLow › HighLowGameActivity
1: Project
7: Structure
Captures
2: Favorites
Build Variants
Gradle
Device File Explorer
Terminal
Build
6: Logcat
3: Find
4: Run
5: Debug
TODO
3 Event Log
59:35 LF÷ UTF-8÷ Context: <no context>

app [~/codeclan/e20/workfiles/projects/CardGame/HighLow/version_4.0/app] - .../src/main/java/com/example/alexs/compendium/HighLow/HighLowGameActivity.java [app]

HighLowGameActivity.java

```java
    public void onPlayButtonClick(View view) {
//Start by resetting hands for both players
        player1.resetPlayerHand();
        player2.resetPlayerHand();
        game.deal();
        winner = game.checkWinner();

        if(winner == player1) {
            winnerName = "Player 1";
        } else if (winner == player2){
            winnerName = "Player 2";
        } else {
            winnerName = "Draw!";
        }
//Get cards from both players and set file images to display the cards
        player1Card1 = player1.getHand().getCards().get(0).getImageFileName();
        player1Card2 = player1.getHand().getCards().get(1).getImageFileName();
        player2Card1 = player2.getHand().getCards().get(0).getImageFileName();
        player2Card2 = player2.getHand().getCards().get(1).getImageFileName();
//Generate results strings
        player1Result = "Player 1" + "\n" + player1.getHand().getCards().get
    (0).prettyName()
                + ", " + player1.getHand().getCards().get(1).prettyName() +
    "\n" + "Score: " + player1.getHand().getHandValue();

        player2Result = "Player 2" + "\n" + player2.getHand().getCards().get
    (0).prettyName()
                + ", " + player2.getHand().getCards().get(1).prettyName()+
    "\n" + "Score: " + player2.getHand().getHandValue();
//Display winner
        displayWinner = winnerName.toUpperCase() + " WINS!!";
//Send results strings through to a new screen where they will be displayed
        Intent intent = new Intent( packageContext: this, HighLowResultActivity
    .class);
        intent.putExtra( name: "player1Card1", player1Card1);
```

HighLowGameActivity › onPlayButtonClick()

HighLowResultActivity.java

```java
        ImageView imageViewp1c1 = findViewById(R.id.imageViewp1c1ID);
        ImageView imageViewp1c2 = findViewById(R.id.imageViewp1c2ID);
        ImageView imageViewp2c1 = findViewById(R.id.imageViewp2c1ID);
        ImageView imageViewp2c2 = findViewById(R.id.imageViewp2c2ID);

        Intent intent = getIntent();
        String player1Card1 = (String) intent.getSerializableExtra
    ( name: "player1Card1");
        String player1Card2 = (String) intent.getSerializableExtra
    ( name: "player1Card2");
        String player2Card1 = (String) intent.getSerializableExtra
    ( name: "player2Card1");
        String player2Card2 = (String) intent.getSerializableExtra
    ( name: "player2Card2");

        String player1Result = (String) intent.getSerializableExtra
    ( name: "player1Result");
        String player2Result = (String) intent.getSerializableExtra
    ( name: "player2Result");
        String displayWinner  = (String) intent.getSerializableExtra
    ( name: "displayWinner");
//Get the names of the files representing the player cards
        int p1c1 = getResources().getIdentifier(player1Card1,
   defType: "drawable", getPackageName());
        imageViewp1c1.setImageResource(p1c1);
        int p1c2 = getResources().getIdentifier(player1Card2,
   defType: "drawable", getPackageName());
        imageViewp1c2.setImageResource(p1c2);
        int p2c1 = getResources().getIdentifier(player2Card1,
   defType: "drawable", getPackageName());
        imageViewp2c1.setImageResource(p2c1);
        int p2c2 = getResources().getIdentifier(player2Card2,
   defType: "drawable", getPackageName());
        imageViewp2c2.setImageResource(p2c2);
//Set the results strings in Results screen to display the winner
```

HighLowResultActivity › onCreate()

Tests Passed: 2 passed (19 minutes ago)

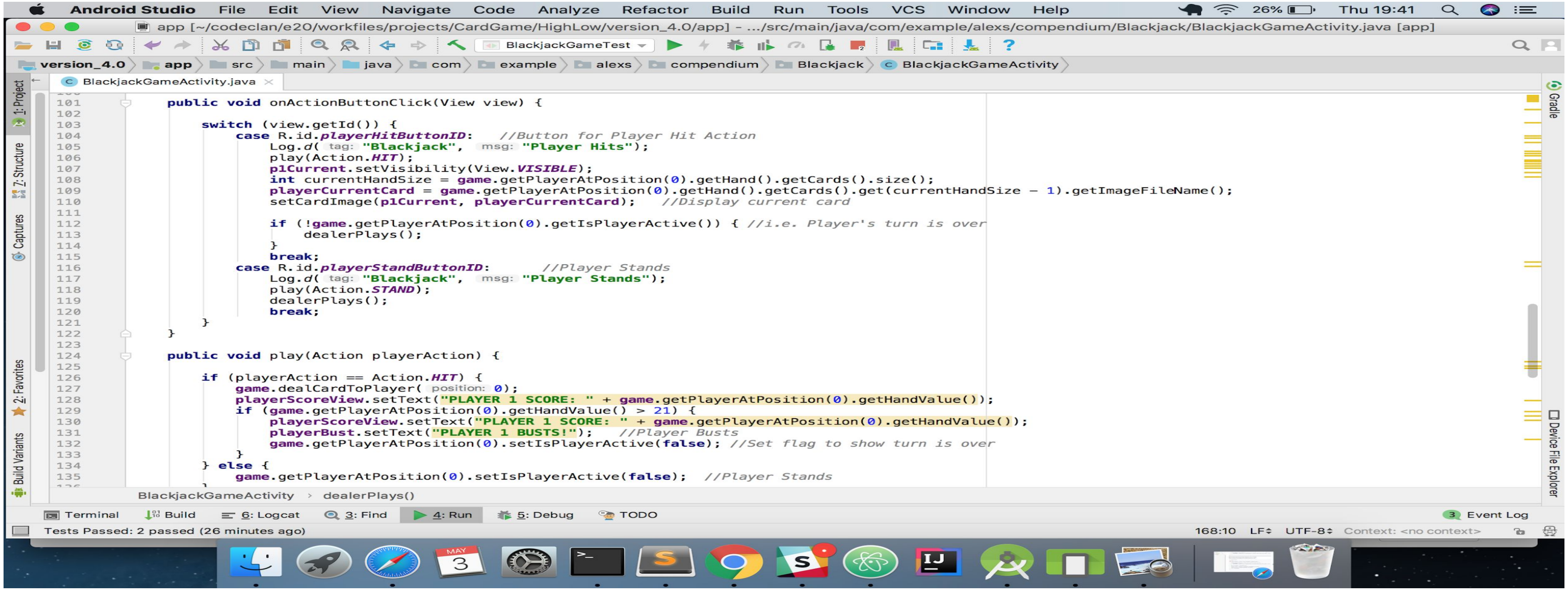
Blackjack Game Activity Screens Java Class

Blackjack Game

Game is run by a Java class called BlackjackGameActivity which sets up the Player, Dealer and Game objects

Controlled by a button which feeds the Player actions (from an Enum called Action) into the game

Blackjack Game Activity Screen: Java Class



app [~/codeclan/e20/workfiles/projects/CardGame/HighLow/version_4.0/app] - .../src/main/java/com/example/alexs/compendium/Blackjack/BlackjackGameActivity.java [app]

```java
    public void onActionButtonClick(View view) {

        switch (view.getId()) {
            case R.id.playerHitButtonID:    //Button for Player Hit Action
                Log.d( tag: "Blackjack",  msg: "Player Hits");
                play(Action.HIT);
                p1Current.setVisibility(View.VISIBLE);
                int currentHandSize = game.getPlayerAtPosition(0).getHand().getCards().size();
                playerCurrentCard = game.getPlayerAtPosition(0).getHand().getCards().get(currentHandSize - 1).getImageFileName();
                setCardImage(p1Current, playerCurrentCard);   //Display current card

                if (!game.getPlayerAtPosition(0).getIsPlayerActive()) { //i.e. Player's turn is over
                    dealerPlays();
                }
                break;
            case R.id.playerStandButtonID:       //Player Stands
                Log.d( tag: "Blackjack",  msg: "Player Stands");
                play(Action.STAND);
                dealerPlays();
                break;
        }
    }

    public void play(Action playerAction) {

        if (playerAction == Action.HIT) {
            game.dealCardToPlayer( position: 0);
            playerScoreView.setText("PLAYER 1 SCORE: " + game.getPlayerAtPosition(0).getHandValue());
            if (game.getPlayerAtPosition(0).getHandValue() > 21) {
                playerScoreView.setText("PLAYER 1 SCORE: " + game.getPlayerAtPosition(0).getHandValue());
                playerBust.setText("PLAYER 1 BUSTS!");   //Player Busts
                game.getPlayerAtPosition(0).setIsPlayerActive(false); //Set flag to show turn is over
            }
        } else {
            game.getPlayerAtPosition(0).setIsPlayerActive(false);  //Player Stands
```

Tests Passed: 2 passed (26 minutes ago)

Happy Moment

Was having problems setting the ImageResource for the card image files, because the instruction looks like:

ImageView imageView:

imageView.setImageResource(<filename>)

and there are 52 filenames, e.g. “c8.png” for Eight of Clubs, so setting the names using Card Enums would have required a large Switch/Case statement

Used id = getResources().getIdentifier().(<filename String>, “drawable”, getPackageName()); to get the integer id for the png file. Then can just use

```
imageView.setImageResource(id);
```

Then wrote a function getImageFileName() in the Card class to return the String filenames for the png files.

Saved a lot of work!

Would like to have had time for:

HorizontalScrollView with a LinearLayout object, so as to dynamically display cards for the Player/Dealer as they are added, rather than setting the image files manually in the code