CS362 Project Group5 Iteration 2 SlapJack Design Document

Risks include having to redesign once implementation has started, having to fit new game into existing framework.

1. Task Assignee: James Gossling

• How does the GameController know that a game is selected and what does it need to do

before it can call match.start( )? Set up the required infrastructure for GameController to

start the match.

Task Details:

* Add string member SLAPJACK to GameFactoryFactory
* Add SLAPJACK to gameIds
* Modify createGameFactory to be able to create SlapJackGameFactory
* Create Class SlapJackGameFactory and implement methods

Task dependencies: None

2. Task Assignee: Cody Tomkins

• How does MatchController know when the game can begin and what does it need to do

before it can call mainloop.play( )? Set up the required infrastructure for MatchController

to start the match.  **-> implement slapjackquorum, slapjackinitcmd and put in slapjackrules**

Task Details:

* MatchController knows it can begin when table.partiesReady() gives a true result. This evaluates as true when a quorum has reached its minimum capacity. To implement this, we would need to create a slapjackQuorum class that would have a minimum and maximum of 2 players. If there are less than two players, the application will wait in the while loop inside of MatchController.
* Before beginning play but after the controller has enough players, the MatchController creates an initcmd (implements the Move interface). We would need to create a new slapjackInitCmd and place it in the slapjackrules. This will call apply to the table and views like before. It then creates the PlayController like normal and begins the game.

Task dependencies: None

3. Task Assignee: Shivam Vashi

• At the start of play there should be a deal button and the title of the game should be set to

“Slapjack”.  **-> implement slapjackinitcmd**

Task Details:

Create a constructor with the players, game, and table for slapjackinitcmd

Create apply commands with table and views and input

Have apply with the view parameter set up the deal button after creating the random and discard piles  
Task dependencies: None

4. Task Assignee: James Gossling

• Pressing the deal button results in two even piles of shuffled face down cards. There are

many simple shuffle algorithms, any is fine. **-> implement slapjackinitcmd & slapjackdealcommand**

Task Details:

* Implement apply(Table table) in class SlapJackInitCmd
* Shuffle 26 cards into p1 pile face down
* Shuffle 26 cards into p2 pile face down
* Create empty center pile for placing cards
* Add piles to table
* Create class SlapJackDealCommand
* Implement apply(Table table) and apply(ViewFacade views)

Task dependencies: Task 1, Task 2

5. Task Assignee: Andrew Pester

• During play, the player’s alternate selecting the card at the top of their pile. Their card is

placed face up on a center pile. Players are ignored if they select a card out of turn or

from a pile that does not belong to them.  **-> implement slapjackrules & slapjackmove**

Task Details:

* Use the existing CardEvent to deal with a card on the table that is clicked. This will work identically to how it is implemented for P52.
* Implement an apply(CardEvent e, Table table, Player player). This will be in the rules for slapjack the rules will check if it is that players turn to place a card on the center deck. apply will also check if they are selecting from the correct pile. If it is that person's turn and if they are selecting from their deck then the apply method will return a new SlapjackMove. If they do not either select from their pile or if they click their pile when it is not their turn return a DropEventCmd.
* Implement a simple constructor that is identical to the P52Move constructor.
* Implement an apply(Table table) for the SlapjackMove that will remove a card from that player's respective pile and place it face up in the center pile. This doesn’t actually handle the player’s views but does update the values that our code has so that we can then update the views.
* Implement an apply(ViewFacade view) inside of SlapjackMove that will send a HideCardRemote to hide the topmost card on that player's pile. Then it will remove the topmost card from the player's pile using HideCardRemote. Then it will add the card to the center pile using AddCardRemote and proceed to show the card in the center pile using ShowCardRemote.

Task dependencies: Task 4

6. Task Assignee: Andrew Pester

• When a player selects a Jack on the center pile all cards in the pile are transferred to the

bottom of their pile. A player that selects a card incorrectly is ignored. **->implement slapjackrules & RoundWinMove**

Task Details:

* When a CardEvent is created and sent to the SlapjackRules the first thing that the apply method from the previous task should do is check if that card that the player clicked is a jack and if it is in the center pile. If both of these are true then it would return a new RoundWinMove command. If either of these conditions are not true then it will return a DropEventCmd
* RoundWinMove command will implement Move and will be much like P52Move. This command will have a constructor, apply(Table table), apply(ViewFacade view).
* The constructor would take the Player, toPile and fromPile.
* apply(Table table) will take all of the cards from the center pile using getCards() in Pile. Then using a loop for all the cards in the center pile it will add them to the player’s pile using addToPile().
* apply(ViewFacade view) will update the views of the players looping through the collection of cards again and hiding each card using HideCardRemote. Then it will remove it from the pile using RemoveFromPileRemote. Then it will add it to the round winner’s pile using AddToPileBottomRemote. Following this it will Show the card using ShowCardRemote. This will happen for every card in the center pile until the round winner has had all the cards added to their deck.

Task dependencies: Task 5

7. Task Assignee: Cody Tomkins

• The display of the players score always represents the number of cards in their pile.

**-> Implement slapjackrules**

Task Details:

* When a CardEvent Occurs, the players score should decrement (since they are putting their card in the center pile)
* Once a player calls slapjack successfully, the “winning” player’s score should increment by the size of the pile.
* Each players score should initially be the amount of starting cards in their deck (this can be done in the slapjackInitCmd class)

Task dependencies: Task 3, Task 6

8. Task Assignee: Shivam Vashi

• When a player runs out of cards the other player wins at the end of their turn. Set the title

of the game to “Player X Wins”. **-> implement slapjackrules**

Task Details:

On an apply event with a card event parameter, check player’s scores and if a player 0 for their score, set the other player as winner and set the title to the other player winning  
Task dependencies: Task 7

9. Task Assignee:

• (optional) Extra game play rule: if a player improperly selects a center card that is not a

Jack the other player wins all of the cards in the center pile. **-> implement slapjackrules**

Task Details:  
Task dependencies:

10. Task Assignee:

• (optional) Extra game play rule: on each play the center pile is moved to a random

location to prevent a player from hovering over the pile. **-> implement slapjackrules?**

Task Details:  
Task dependencies:

11. Task Assignee:

• (optional) When the game is finished show the deal button. This will be helpful.

String remoteId = view.getRemoteId(DealButton.kSelector);

view.send(new ShowButtonRemote(remoteId)); **-> implement slapjackrules?**

Task Details:  
Task dependencies:

12. Task Assignee:

• (optional) When the deal button is selected for a new game, deal the existing cards (not

new ones) to the two players. **-> implement slapjackrules?**

Task Details:  
Task dependencies:

NOTES:

Need to change gamefactoryfactory to be able to choose slapjackfactory

Slapjack package:

Slapjackgamefactory

Slapjackrules -> Need to be turnbased

Slapjackdealcommand

Slapjackinitcmd -> sets up card piles etc

Slapjackmove

Slapjackquorum

RoundWinMove

can reuse:

from p52 package:

P52player

Partyrole

Setquorumcmd

P52playerview

Createplayercmd

Dealbutton

endplaymove

From model package:

Tablebase