Setup:

- select number of player
- create cave and dragon token for player
- create each cut and uncut volcano card that consist of cut tile and uncut tile
- create chit card and randomise.

Flipping chit card:

- player select a chit card and the chit card will be flipped
- if player select a correct card and he can select another card again
- if player select wrong card uncover all the chit card and the player end the tu
- if player select dragon pirate, player still can select another card

Movement of dragon tokens based on their current position as well as the last flippe
card

- player move forward with corresponding number of animal if he choosing the
- player move backward if he choosing the dragon pirate
- If a player's dragon would land on an occupied square, they cannot move it, a

Change of turn to the next player:

- if player select a correct card and he can select another card again
- if player select wrong card the player end the turn
- if player select dragon pirate, player still can select another card

Winning the game:

- when the player move the exact of number to the cave the game win.
- A victory message is displayed for the winning player.

rn

d dragon

 correct chit card

and their turn ends

**Game**

- players: [player]
- gameBoard: GameBoard
- playerWinning: False

- select_player_num(int)
- create_player(int)
- create_game_board(player: Pla
- run()
- process_player_turn()
- initialise_game()

**CutVolcanoCard**

yer)

## CutVolcanoCard

-tiles: [Tile]

---

+ add_tile(tile: Tile)
+ get_tiles()

---

## <<Abstract>>
## VolcanoCard

-tiles: [Tile]

---

+ add_tile(tile: Tile)
+ get_tiles()

Extends

Extends

## UncutVolcanoCard

-tiles: [Tile]

---

+ add_tile(tile: Tile)
+ get_tiles()

## GameBoard

- occupiable_item: [Occupiable]
- chit_cards: [ChitCard]

---

- create_cave(player: Player)
- create_dragon_token(player: Pla
- create_volcano_card()
- create_chit_card()
- randomise_chit_card()

## <<interface>>
## occupiable

+ occupied(dragon_token:DragonToken)
+ add_dragon_token()
+ remove_dragon_token()

## CutTile

- animal: Animal
- dragon_token: DragonToken
- occupied: boolean

---

+ occupied(dragon_token:DragonToken)
+ add_dragon_token()
+ remove_dragon_token()

Extends

## <<Abstract>>
## Tile

- animal: Animal
- dragon_token: DragonToken
- occupied: boolean

---

+occupied(dragon_token:DragonToken)
+ add_dragon_token()
+ remove_dragon_token()

## Cave

- animal: Animal
- dragonToken: DragonToken
- dragonBack: Boolean
- player_id: int

---

+ occupied(dragon_token:Drago
+ add_dragon_token()
+ remove_dragon_token()

Extends

## UncutTile

- animal: Animal
- dragon_token: DragonToken
- occupied: boolean

---

+ occupied(dragon_token:DragonToken)
+ add_dragon_token()
+ remove_dragon_token()

## DragonToken

-current_position_anima

---

+ move_forward(int)
+ move_backward(int)

## <<enumeration>>
## Animal

BABY_DRAGON
SPIDER
SALAMANDER
BAT
DRAGON_PIRATE

## Player

id: int
cave: Cave
dragon_token: DragonToken

+ add_cave(cave: Cave)
+ add_dragon_token(dragon_token: Dragon)
+ play_turn(occupiable_item, chit_cards)
+ move_dragon_token(occupiable_item)
+ flip_chit_card(chit_cards)
+ get_id()

## ForwardChitCard

+ animal: Animal
+ animal_quantity: int
+ flipped: boolean

+ flip()
+ cover()
+ is_flipped()
+ return_step(Animal)

## <<Abstract>>
## ChitCard

- animal: Animal
- animal_quantity: int
- flipped: boolean

+ flip(animal: Animal)
+ cover()
+ is_flipped()

Extends

## BackwardChitCard

+ animal: Animal
+ animal_quantity: int

+ flip()
+ cover()
+ is_flipped()
+ return_step(Animal)

Extends