



Real Estate Game

Knox Game Design

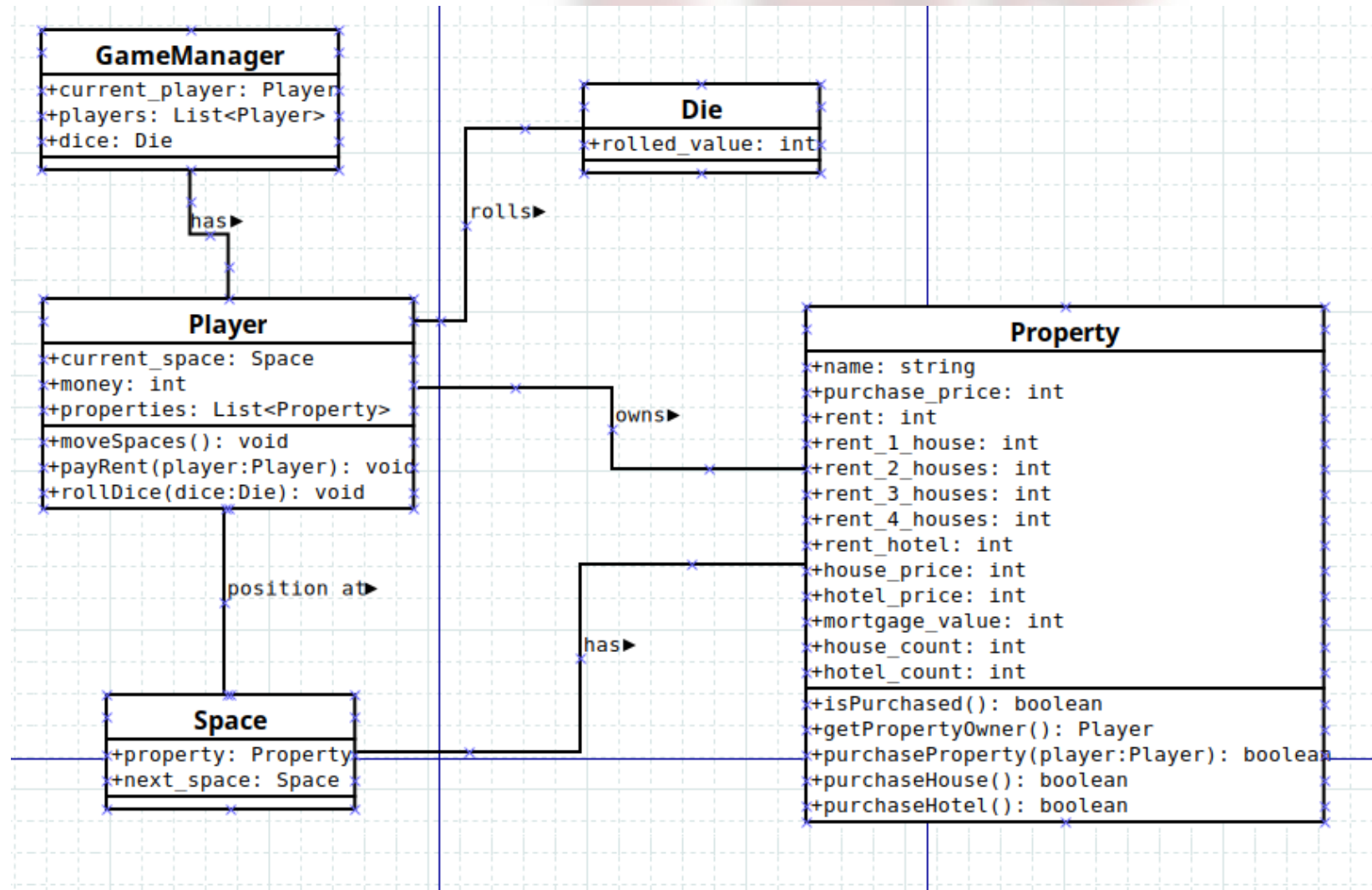
December 2025

Overview

- Not for beginner programmers
 - Multiple states and game loops / modes
 - “Build one to throw away” – Fred Brooks, Mythical Man Month
- Create list of rules / requirements
- Get basic gameplay working first. Then work on graphical display and interface.
- Can design/implement in any order. I’m just offering the steps that I used.
- Design will probably be revised during implementation

KNOX
GAME
DESIGN

Software Design – Class Diagram

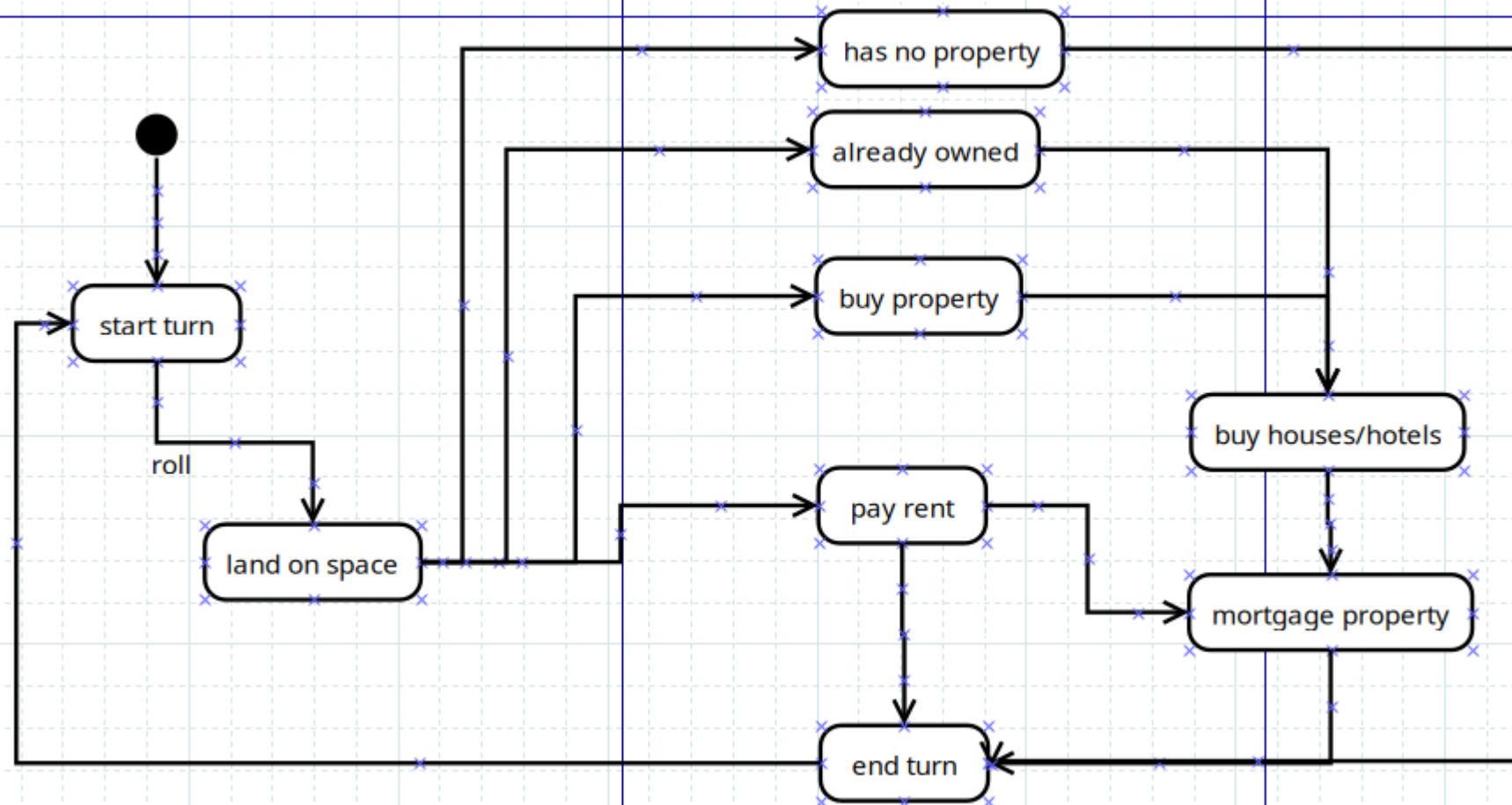


KNOX
GAME
DESIGN

Design decisions

- Linked list of spaces
 - Array would have been more complex
- Purchase methods return boolean (success / failure)
 - Could use exceptions / error handling
- Player has reference to their current space
 - Spaces could have reference to collection of Players on that space
- Player has collection of purchased Property
 - Could have Property have reference to owned Player
 - Issue – multiple players could own same property if not handled correctly
 - Makes looping through Player's owned properties easier
 - Finding the property's owner requires looping through all of the players properties
 - Avoid having player property List AND reference to owner player in Property. Do one or the other, but NOT BOTH.
 - Make getPlayerOwner method in Property to return the property's owner

Software Design – State Diagram



KNOX
GAME
DESIGN

Step 1 – Moving players around the board

- Load properties list from file
- Create GameManager, Player, Space, Property, and Die classes
- Display objects to screen
- Randomize dice value when 'R' pressed
- Move current player number of spaces rolled
- Set current player to next player when turn ends
- Only do enough graphical display to convey current game status

0:	10:	20:	30:
1: Oliver Springs	11: Coalfield	21: Lenior City	31: Alcoa
2:	12:	22:	32: Maryville
3: Karns	13: Wartburg	23: Morristown	33:
4:	14: Sunbright	24: Seymour	34: Farragut
5:	15:	25:	35:
6: Kingston	16: Pigeon Forge	26: Clinton	36:
7:	17:	27: Powell	37: Oak Ridge
8: Rockwood	18: Gatlinburg	28:	38:
9: Harriman	19: Halls Crossroads	29: Sevierville	39: Knoxville

Current player: P5
R: Roll
Dice: 3, 4

1	
2	Oliver Springs
3	
4	Karns
5	
6	
7	Kingston
8	
9	Rockwood
10	Harriman
11	
12	Coalfield
13	
14	Wartburg
15	Sunbright
16	
17	Pigeon Forge
18	
19	Gatlinburg
20	Halls Crossroads
21	
22	Lenior City
23	
24	Morristown
25	Seymour
26	
27	Clinton
28	Powell
29	
30	Sevierville
31	
32	Alcoa
33	Maryville
34	
35	Farragut
36	
37	
38	Oak Ridge
39	
40	Knoxville

GAME
DESIGN

Step 2 – Money / Property Purchase / States

- Give each player an initial amount of game money
 - Design decision – money tracked as integers. not individual bills
- Add cost of each property in data file
 - Design decision – using CSV (comma separated values) format. Could translate to JSON or YAML, but extra parsing code is required.
- Allow player to buy unowned property when stopping on it
 - Player money must be greater than purchase cost
 - Subtract purchase cost from player money
 - Assign instance of property to player
- Add GameState enum
 - StartTurn
 - LandOnSpace
 - EndTurn

0:	10: Coalfield \$140	20: Lenior City \$220	30: Alcoa \$300
1: Oliver Springs \$60	11:	21:	31: Maryville \$300
2:	12: Wartburg \$140	22: Morristown \$220	32:
3: Karns \$60	13: Sunbright \$160	23: Seymour \$240	33: Farragut \$320
4:	14:	24:	34:
5: Kingston \$100	15: Pigeon Forge \$180	25: Clinton \$260	35:
6:	16:	26: Powell \$260	36: Oak Ridge \$350
7: Rockwood \$100	17: Gatlinburg \$180	27:	37:
8: Harriman \$120	18: Halls Crossroads \$200	28: Sevierville \$280	38: Knoxville \$400
9:	19:	29:	39:

Current player: P4	P1	\$660
Dice: 3, 6	P2	\$1240
B: Buy Property	P3	\$1160
E: End Turn	P4	\$960
	P5	\$740
	P6	\$1150


Property color is the color of owning Player

1	
2	Oliver Springs, 60
3	
4	Karns, 60
5	
6	Kingston, 100
7	
8	Rockwood, 100
9	Harriman, 120
10	
11	Coalfield, 140
12	
13	Wartburg, 140
14	Sunbright, 160
15	
16	Pigeon Forge, 180
17	
18	Gatlinburg, 180
19	Halls Crossroads, 200
20	
21	Lenior City, 220
22	
23	Morristown, 220
24	Seymour, 240
25	
26	Clinton, 260
27	Powell, 260
28	
29	Sevierville, 280
30	
31	Alcoa, 300
32	Maryville, 300
33	
34	Farragut, 320
35	
36	
37	Oak Ridge, 350
38	
39	Knoxville, 400

KNOX
GAME
DESIGN

Step 3 – Rent / Eliminate Player / Game Over

- Add rent values to property data file
- Transfer rent value to property owner when landing on property space
 - Display message stating amount transferred
- Player is eliminated if unable to pay rent
 - Remove player from GameManager players List
 - Win condition – only one player left in players List
 - Transition to GameOver state
 - Test by making rent a really high value (all rents \$5000)
 - Remember to set it back after testing
 - When player eliminated, all of their properties become available again, which may not follow “traditional” rules

0:		10: Coalfield	20: Lenior City	30: Alcoa
		\$140	\$220	\$300
1: Oliver Springs		11:	21:	31: Maryville
\$90				\$300
2:		12: Wartburg	22: Morristown	32:
		\$140	\$220	
3: Karns		13: Sunbriht	23: Seymour	33: Farragut
\$90		\$160	\$240	\$320
4:		14:	24:	34:
5: Kingston		15: Pigeon Forge	25: Clinton	35:
\$100		\$180	\$260	
6:		16:	26: Powell	36: Oak Ridge
			\$260	\$350
7: Rockwood		17: Gatlinburg	27:	37:
\$100		\$180		
8: Harriman		18: Halls Crossroads	28: Sevierville	38: Knoxville
\$120		\$200	\$280	\$480
9:		19:	29:	39:

P5 paid \$16 to P6 at Halls Crossroads

0:	10: Coalfield	20: Lenior City	30: Alcoa
	\$140	\$230	\$350
1: Oliver Springs	11:	21:	31: Maryville
\$60			\$300
2:	12: Wartburg	22: Morristown	32:
	\$140	\$220	
3: Karns	13: Sunbright	23: Seymour	33: Farragut
\$80	\$160	\$240	\$320
4:	14:	24:	34:
5: Kingston	15: Pigeon Forge	25: Clinton	35:
\$100	\$180	\$260	
6:	16:	26: Powell	36: Oak Ridge
		\$280	\$330
7: Rockwood	17: Gatlinburg	27:	37:
\$100			
8: Harriman	18: Halls Crossroads	28: Sevierville	38: Knoxville
\$120	\$200	\$300	\$400
9:	19:	29:	39:

P3 unable to pay \$5000 at Wartburg. Eliminated from game.

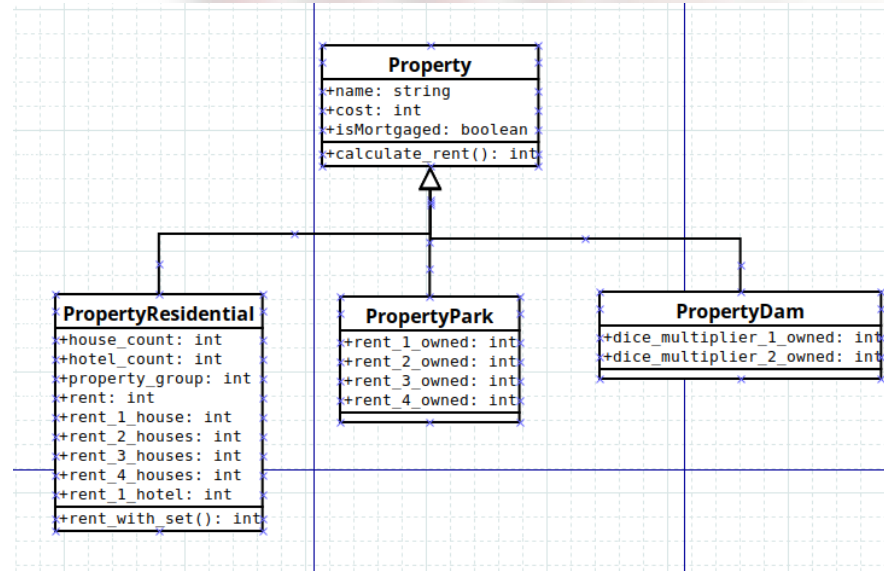
Game Over: Player P3 wins!

1	
2	■ Oliver Springs, 60, 2
3	
4	■ Karns, 60, 4
5	
6	■ Kingston, 100, 6
7	
8	■ Rockwood, 100, 6
9	■ Harriman, 120, 8
10	
11	■ Coalfield, 140, 10
12	
13	■ Wartburg, 140, 10
14	■ Sunbright, 160, 12
15	
16	■ Pigeon Forge, 180, 14
17	
18	■ Gatlinburg, 180, 14
19	■ Halls Crossroads, 200, 16
20	
21	■ Lenior City, 220, 18
22	
23	■ Morristown, 220, 18
24	■ Seymour, 240, 20
25	
26	■ Clinton, 260, 22
27	■ Powell, 260, 22
28	
29	■ Sevierville, 280, 24
30	
31	■ Alcoa, 300, 26
32	■ Maryville, 300, 26
33	
34	■ Farragut, 320, 28
35	
36	
37	■ Oak Ridge, 350, 35
38	
39	■ Knoxville, 400, 50

The logo for Knox Game Design features the word "KNOX" in red, "GAME" in green, and "DESIGN" in blue, all in a pixelated, blocky font. The text is arranged in three lines, with "DESIGN" being the largest and most prominent.

Step 4 – Ownable Properties / Calculating Rent

- 22 “residential”
 - 8 sets
 - can have 4 houses each
 - can have 1 hotel each
- 4 “parks”
 - rent is based on number owned (1x, 2x, 4x, 8x)
- 2 “dams”
 - rent is based on dice roll and number owned (4x, 10x)
- calculate_rent is abstract method that must be implemented by subclass
- testing requires controlling roll values to land on test case spaces (park/dam). using random rolls would take forever to land on test case spaces
- mortgage_value is actually a calculated value: half the purchase price



Rent on PropertyPark

```
2 references
public override int calculateRent() {
    int iParkCount;
    iParkCount = getPropertyOwner().getOwnedParkCount();
    return 25 * ((int) MathF.Pow(2, iParkCount - 1));
}
```

Rent on PropertyDam

```
3 references
public override int calculateRent() {
    int iDamCount;
    int iMultiplier;
    iDamCount = getPropertyOwner().getOwnedDamCount();

    switch(iDamCount) {
        case 1:
            iMultiplier = 4;
            break;
        case 2:
            iMultiplier = 10;
            break;
        default:
            return 0;
    }

    return (gamemanager.dice[0].iRolledValue + gamemanager.dice[1].iRolledValue) * iMultiplier;
}
```

0:	10: Coalfield Buy \$140	20: Lenior City Buy \$220	30: Alcoa Buy \$300
1: Oliver Springs Buy \$80	11: Watts Bar Dam Buy \$150	21:	31: Maryville Buy \$300
2:	12: Wartburg Buy \$140	22: Morristown Buy \$220	32:
3: Karns Buy \$60	13: Sunbright Buy \$160	23: Seymour Buy \$240	33: Farragut Buy \$320
4: A.K. Bissell Park Rent \$200	14: Chilhowee Park Rent \$200	24: Market Square Rent \$200	34: World's Fair Park Rent \$200
5: Kingston Buy \$100	15: Pigeon Forge Buy \$160	25: Clinton Buy \$200	35:
6:	16:	26: Powell Buy \$200	36: Oak Ridge Buy \$350
7: Rockwood Buy \$100	17: Gatlinburg Buy \$180	27: Norris Dam Buy \$150	37:
8: Harriman Buy \$120	18: Halls Crossroads Buy \$200	28: Sevierville Buy \$280	38: Knoxville Buy \$400
9:	19:	29:	39:

Current player: P2 P1 \$1075 P2 paid \$200 to P1 at A.K. Bissell Park.
Dice: 2, 3 P2 \$1125

KNOX
GAME
DESIGN

- Add property set id to Residential properties
- Update properties data file with remaining property values
 - Update PropertyResidential calculateRent method to doubled rent if all properties in set owned by the same player
- Update parser to create Residential/Park/Dam based on first CSV token on line

RealEstate			
	Coalfield Buy\$140	Lenior City Rent\$36 Set owned	Alcoa Rent\$26
Oliver Springs Buy\$60	Watts Bar Dam Buy\$150		Maryville Rent\$26
	Wartburg Rent\$10	Morristown Rent\$36 Set owned	
Karns Rent\$4	Sunbright Rent\$12	Seymour Rent\$40 Set owned	Farragut Buy\$320
A.K. Bissell Park Buy\$200	Chilhowee Park P1 Rent\$25	Market Square Rent\$25	World's Fair Park Buy\$200
Kingston Rent\$12 Set owned	Pigeon Forge Buy\$180	Clinton Rent\$44 Set owned	
		Powell Rent\$44 Set owned	Oak Ridge P2 Buy\$350
Rockwood Rent\$12 Set owned	Gatlinburg Rent\$14	Norris Dam P3 Rent\$16	
Harriman Rent\$16 Set owned	Halls Crossroads Rent\$16	Sevierville Rent\$48 Set owned	Knoxville Buy\$400
Current player: P1 P1 \$196 P1 purchased Chilhowee Park Dice: 2, 2 P2 \$350 P3 \$264 E: End Turn			

1	
2	R,0,Oliver Springs,60,2,10,30,90,160,250,50,50
3	
4	R,0,Karns,60,4,20,60,180,320,450,50,50
5	P,A.K. Bissell Park,200
6	R,1,Kingston,100,6,30,90,270,400,550,50,50
7	
8	R,1,Rockwood,100,6,30,90,270,400,550,50,50
9	R,1,Harriman,120,8,40,100,300,450,600,50,50
10	
11	R,2,Coalfield,140,10,50,150,450,625,750,100,100
12	D,Watts Bar Dam,150
13	R,2,Wartburg,140,10,50,150,450,625,750,100,100
14	R,2,Sunbright,160,12,60,180,500,700,900,100,100
15	P,Chilhowee Park,200
16	R,3,Pigeon Forge,180,14,70,200,550,750,950,100,100
17	
18	R,3,Gatlinburg,180,14,70,200,550,750,950,100,100
19	R,3,Halls Crossroads,200,16,80,220,600,800,1000,100,100
20	
21	R,4,Lenior City,220,18,90,250,700,875,1050,150,150
22	
23	R,4,Morristown,220,18,90,250,700,875,1050,150,150
24	R,4,Seymour,240,20,100,300,750,925,1100,150,150
25	P,Market Square,200
26	R,5,Clinton,260,22,110,330,800,975,1150,150,150
27	R,5,Powell,260,22,110,330,800,975,1150,150,150
28	D,Norris Dam,150
29	R,5,Sevierville,280,24,120,360,850,1025,1200,150,150
30	
31	R,6,Alcoa,300,26,130,390,900,1100,1275,200,200
32	R,6,Maryville,300,26,130,390,900,1100,1275,200,200
33	
34	R,6,Farragut,320,28,150,450,1000,1200,1400,200,200
35	P,World's Fair Park,200
36	
37	R,7,Oak Ridge,350,35,175,500,1100,1300,1500,200,200
38	
39	R,7,Knoxville,400,50,200,600,1400,1700,2000,200,200

Step 5 – Mortgage / Unmortgage

- Need a separate game loop for selecting properties to mortgage / unmortgage
- Mortgage value is half the purchase cost – use method instead of storing in variable
- Update rent collection code to not charge when landing on mortgaged property
- Allow unmortgage if players money is greater than or equal to mortgage value * 1.1

Select property to mortgage

A.K. Bissell Park
M Clinton
Alcoa
M World's Fair Park
Kingston
M Sunbright
M Lenior City

Oliver Springs Buy \$60	Coalfield Buy \$140	Lenior City Mortgaged	Alcoa Rent \$20
Karns Buy \$60	Watts Bar Dam Buy \$150	Morristown Buy \$220	Maryville Buy \$300
A.K. Bissell Park Rent \$50	Wartburg Buy \$10	Seymour Rent \$20	Farragut Rent \$20
Kingston Rent \$2	Sunbright Mortgaged	Market Square Buy \$200	World's Fair Park Mortgaged
Rockwood Rent \$9	Chilhowee Park Buy \$200	Clinton Mortgaged	Oak Ridge Rent \$35
Harriman Rent \$9	Pigeon Forge Buy \$180	Powell Buy \$200	Knoxville Mortgaged
	Gatlinburg Mortgaged	Norris Dam Buy \$150	
	Halls Crossroads Rent \$16	Sevierville Mortgaged	
Current player: P3			
M: Mortgage			
U: Unmortgage			
Q: Return			

P3 purchased Lenior City

Select property to mortgage
A.K. Bissell Park
M Clinton
Alcoa
M World's Fair Park
Kingston
M Sunbright
M Lenior City

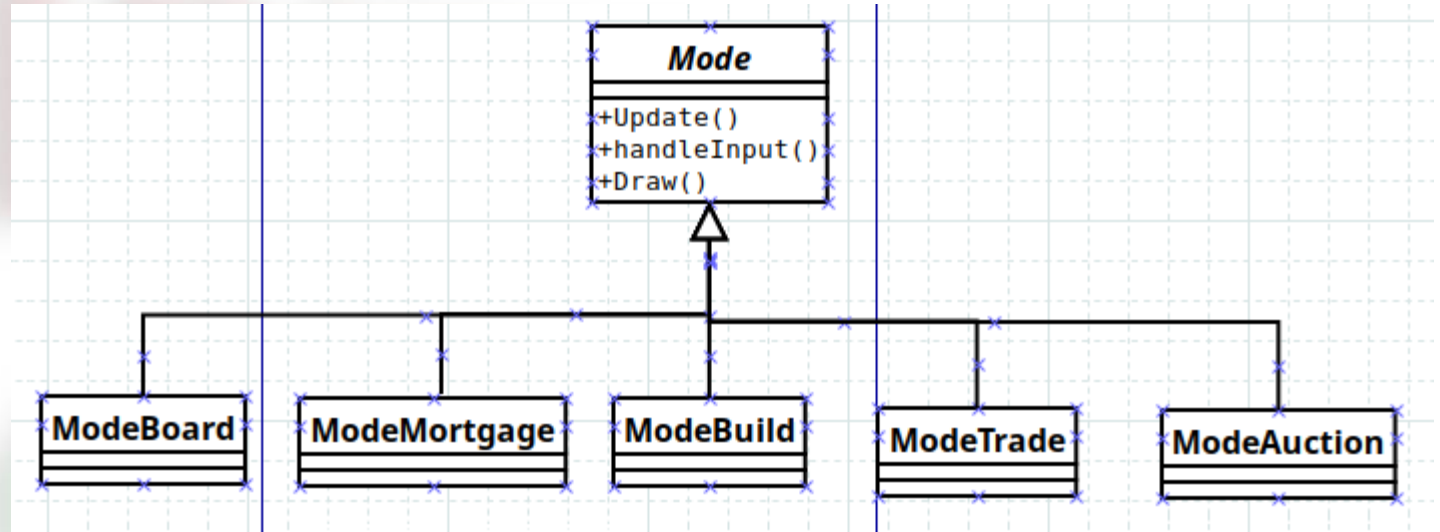
Clinton
Mortgaged

Clinton is mortgaged. No rent paid

KNOX
GAME
DESIGN

Step 6 - Game Modes / Building Houses and Hotels

- Modes to separate distinct functions
- Track current mode in the GameManager
- Modes implement Update, and Draw methods
- Build mode
 - Allow player to buy houses and hotels if they own all properties in set
 - Remember to recalculate rent value based on house / hotel count
 - Must check that property class type is PropertyResidential, then cast it to use the house / hotel properties



Coalfield Rent \$750 Hotels: 1
Oak Ridge Rent \$1300 Houses: 4

Select property to build		
A.K. Bissell Park		
Coalfield	Houses: 0	Hotels: 1
Pigeon Forge	Houses: 3	Hotels: 0
Clinton	Houses: 1	Hotels: 0

KNOX
GAME
DESIGN

Step 7 - Trade

- Select player to trade with
 - Build list of properties to trade for each player
 - Allow specification of cash amount
 - Add Yes / No option for both traders
 - Return to trading loop to make changes if one player chooses No
 - Resolve when both players select Yes or trade is cancelled
 - Object oriented makes moving properties between players easy
 - Just move the reference (pointer) from one player to the other
- Trade screen is a “project in itself”
 - Having modes allows development on trade screen separate from the rest of the game

Select player to trade

P1

P2

P3

P2

T Sunbright

Lenior City

T Farragut

Cash \$552

Yes

No

P1

Coalfield

T Seymour

Norris Dam

T World's Fair Park

Watts Bar Dam

T Halls Crossroads

T Morristown

Oak Ridge

Cash \$0

Yes

No

KNOX
GAME
DESIGN

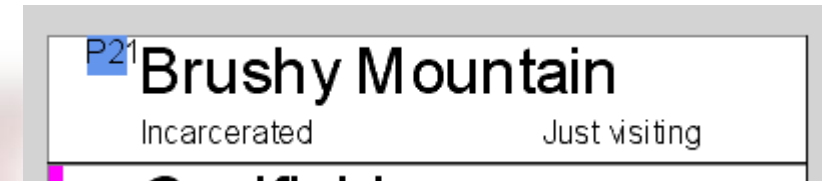
Step 8 - Auction

- Allow property to be auctioned when player lands on property space
- Auction is its own mode and interface
- Set countdown timer for auction to complete
 - Reset countdown after a player bids
- Assign property to player with highest bid
- Design decisions
 - 10 seconds to close after last bid
 - Starting bid is $0.10 \times \text{purchase price}$
 - Next bid is $1.20 \times \text{previous bid}$
 - These values can be modified as needed

Auction: Chilhowee Park		
Next Bid: 276		Countdown: 5
P1	134	
P2	230	
P3	192	

Step 9 – Doubles, Arrested, Incarceration

- Allow player to roll again on doubles if not incarcerated
- Make Incarceration space a subclass of Space class
 - Incarceration space has a List/collection of incarcerated players
- Move player to Incarceration space when landing on the Arrested space
- Track count of times player has rolled doubles in a row
 - Send to incarceration if doubles rolled three times
- Incarcerated options
 - Roll for doubles: `die1.value == die2.value`
 - Pay: `currentPlayer.iMoney -= 50`
 - Card: `currentPlayer.iGetOutCard--`
 - Add instance variable to Player to track GetOut cards
- Track whether player is incarcerated or “just visiting”
- Incarcerated loop is separate branch from move spaces loop
- Track number of incarcerated rolls



Current player: P2

R: Roll for doubles

P: Pay \$50

C: Use GetOut card

KNOX
GAME
DESIGN

Step 10 – Event Cards

- Event card space a subclass of space
- New EventCard class
- For each card, implement an EventCard subclass
- Add cards to two card lists based on event type
- Randomize the order of the two Lists of event cards
- Draw an EventCard when landing on an EventCard space based on EventCard type
- Implement abstract *action* method on the EventCard class to perform card's instruction in game
 - Good example of using polymorphism in a game
- Only implemented three cards of each type for this demo

Step 11 – Other spaces

- Start – Earn \$200 when landing or passing
- Sunsphere – Free place to park. No action for this space
- Taxes – Pay the required tax amount

Mode: board			
Start	Brushy Mountain <small>Incarcerated Just visiting</small>	Sunsphere	Arrested
Oliver Springs <small>Buy \$60</small>	Coalfield <small>Buy \$140</small>	Lenior City <small>Buy \$220</small>	Alcoa <small>Buy \$300</small>
Mystery Vault	Watts Bar Dam <small>Buy \$150</small>	Happenstance	Maryville <small>Buy \$300</small>
Karns <small>Buy \$60</small>	Wartburg <small>Buy \$140</small>	Morristown <small>Buy \$220</small>	Mystery Vault
Taxes \$200 <small>P1</small>	Sunbright <small>Buy \$160</small>	Seymour <small>Buy \$240</small>	Farragut <small>Buy \$320</small>
A.K. Bissell Park <small>Buy \$200</small>	Chilhowee Park <small>Buy \$200</small>	Market Square <small>Buy \$200</small>	World's Fair Park <small>Buy \$200</small>
Kingston <small>Buy \$100</small>	Pigeon Forge <small>Buy \$180</small>	Clinton <small>Buy \$260</small>	Happenstance
Happenstance	Mystery Vault	Powell <small>Buy \$260</small>	Oak Ridge <small>Buy \$350</small>
Rockwood <small>Buy \$100</small>	Gatlinburg <small>Buy \$180</small>	Norris Dam <small>Buy \$150</small>	Taxes \$75
Harriman <small>Buy \$120</small>	Halls Crossroads <small>Buy \$200</small>	Sevierville <small>Buy \$260</small>	Knoxville <small>Buy \$400</small>
Current player: P1		P1 \$2050 2	P1 paid \$200 in taxes
Dice: 2, 3		P2 \$2200 1	
		P3 \$2400 3	

KNOX
GAME
DESIGN

Step 12 – Display Manager

- Update graphics / model view controller
 - Use Display manager and display objects for drawing
 - Model game objects should not have drawing methods
 - Separate the “model” from the “view”
 - Display manager has reference to game manager with references to all game objects
 - Allows switching display presentations (text, 2D, 3D, etc)
- Set target position for moving players
 - Start by just using a linear interpolation from current space to next space
 - This is “good enough” for this demo
 - Add DisplayPlayerToken class to separate token display from player model data
 - Current Player model position will be “ahead” of the token display position. Token display will “catch up” with the model
 - To avoid cutting corners when moving, a list of spaces moved will need to be tracked

SUNSPHERE	LEHIGH CITY RENT \$18	HAPPENING	MORRISTOWN RENT \$18	SEYMOUR RENT \$20	MARKET SQUARE RENT \$50	CLINTON RENT \$22	POWELL RENT \$22	NORRIS DAM RENT \$28	SEVIERVILLE RENT \$24	ARRESTED
HALLS CROSSROADS RENT \$80 SET OWNED HOUSES: 1										ALCOA RENT \$26
GATLINBURG RENT \$70 SET OWNED HOUSES: 1										FRANKFORD RENT \$26
MYSTERY										MYSTERY
PIGEON FORGE RENT \$200 SET OWNED HOUSES: 2	P1	\$924	GOF 1							FARRAGUT RENT \$28
CHILHOWEE PARK RENT \$25	P2	\$1156	GOF 2							WORLD'S FAIR FARM RENT \$25
SUNBRIGHT RENT \$12	P3	\$301	GOF 0							HAPPENING
WARTBURG RENT \$10	P4	\$692	GOF 0 INCARCERATED							DAK RIDGE RENT \$35
WATTS BAR DAM RENT \$28	P5	\$1066	GOF 2							TAXES \$75
COALFIELD BUY \$140	P6	\$611	GOF 1							KNOXVILLE RENT \$50
BRUSHY MOUNTAIN INCARCERATED JUST VISITING										START
HARRIMAN RENT \$8	ROCKWOOD RENT \$6	HAPPENING	KINGSTON RENT \$6	BISSELL PARK RENT \$50	TAXES \$200	WATTS RENT \$4	MYSTERY	OLIVER SPRINGS RENT \$2		

P3 PAID \$28 TO P1 AT WATTS BAR DAM

GAME
DESIGN

What else?

- Finish event cards
- Add sound effects and music
- Allow player to sell / mortgage properties to pay debt (another loop)
- Require player to build houses / hotels evenly between property set