Component 1 - Platforms:

Platforms must have:

* Rect for a physical form.
* Platform method is fed X and Y coordinates as well as length and width on construction to not only build the platform but feed collider info to the level manager for use by the player.
* Level Manager compares the location of the player to each platform on every frame.

How did we execute this?  
- Platform constantly checks for the location of the player relative to itself, flags if the player is directly above it, directly below it, to its left, to its right, or on it.

- If the player lands on the platform, the player stops falling and its Y coordinate is constantly reset to just above the platform.

- If the player hits the bottom of a platform, same thing, just resetting below the platform.

- If the player hits the side of a platform, they stop moving in that direction.

- If no platforms flag the player as being on a platform, then the player is allowed to continue falling.

Problems Encountered:

- Bug Fixing and getting the platforms to a workable level has been a nightmare, a lot of time was spent just getting things to work.

- Even in the final iteration the platforms still break if you jump on them wrong, or at random.

- Platforms being next to each other causes collision problems.

How could we do it better in the future?:

- Ditch resetting the player on contact, just halt their movement next time.

- Instead of basing the platform construction off of rects, have the platforms be a shape drawn using four different vectors that form a circle.

- Should allow for more precise position checking and eliminate the jank that stems from the current position checking model.

Component 2 - Levels

Levels must have:

* An array of platform objects.
* An end goal to trigger a level transition. (If the player collided with the end, go to the next level.
* Optional, a key to remove some kind of barrier from the level. Level check if the key has been collected and toggles the barrier if it has been collected.

How did we execute this?:

- A DrawLevel function uses an array to iterate through every platform in the platform array and uses its DrawPlatform function to draw the platform.

- The level takes the players’ X and Y coordinates and compares them to the location and bounds of the clear zone. If the player is in the clear zone, set the playerInGoal flag in the main method to true so that the level can be iterated.

- The level constructor takes a vector that serves as the spawn and reset positon for the player.

- The key functionality was not implemented, time was short.

Problems encountered:

- Getting the ClearZoneCheck method to work properly took a bit of time but apart from that coding this was actually really smooth.

How could we do it better in the future?:

* ClearZoneCheck could be shuffled to the player.
* This method doesn’t need much changing beyond that.

Component 3 - Player

Player must have:

* A rect for physical form.
* Player dimensions for the rect and bounds checking.
* Vector for position tracking.
* Vector for velocity.
* Booleans for checking if the player is grounded or not as well as eligibility to air dash/double jump.

How did we execute this?:  
- Vectors for the player’s position, velocity and reset point.

- Constant gravity.

- Booleans for jumping, double jumping and left/right movement.

- A boolean to check if the player is grounded.

- PlayerCollisionChecker iterates through all the platforms in the level to check the position of the player to the platforms.

- PlayerBoundsChecker resets the player if they go off the sides of the window or if it falls through the floor.

- PlayerPhysicsHandler uses bool flags to handle where the player is supposed to be going.

- ResetPlayer resets the position of the player according to the level’s respawn point.

Problems Encountered:

- PlayerPhysicsHandler had to be adjusted as time progressed, redundant code had to be deleted.

How could we do this better in the future?:  
- I feel like the PlayerCollisionChecker could be relocated to the Platform class but I’m not so sure.

Component 4 - Game Manager:

The Game Manager must have:

* An array of levels.
* A player object.
* A timer.
* The level manager cycles through each of the platforms in the active level to check collisions with the player.

How did we execute this?:  
- Each level has its own dedicated method to construct the level, this level is then added to the Game Manager’s level array in its dedicated spot.

- Timer not implemented due to text method limitations.

- keyPressed and keyReleased methods handle player controls and game resetting.

- Draw function either draws the start menu, end menu or the game itself depending on a game start flag and how many levels have been completed.

Problems encountered:

- Making the levels was incredibly tedious, adding or removing platforms after the fact required a lot of back and fourth.

How could we do this better in the future?:  
- Level creation could be optimized, more arrays less hard coded variables.

- keyPressed and keyReleased functions directly tied to the player could be moved to the Player class.

- playerInGoal checkers could be moved to the level or player class.