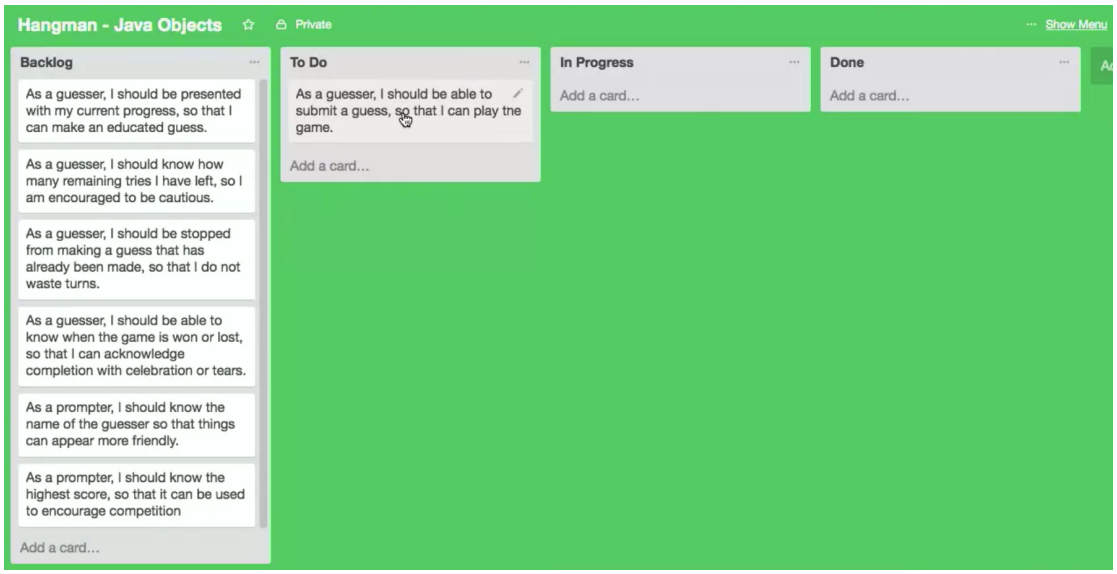
Java Objects Part 3 Notes

Team Treehouse

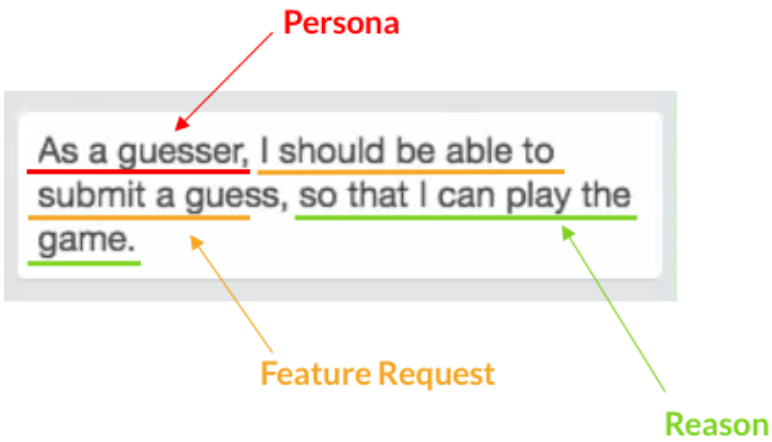
May 23, 2020

1 Planning the MVP

- Priortized backlog of user stories ⇒ a common way of handling scope
  - Uses Kanban board



- Common format of a user story



- Sprint ⇒ Set period of time a list during which work will be completed and will be ready for review (i.e. By the end of the day, dun dun dun...)

2 Quiz 1

1. The set of software development practices we talked about exploring is called

   A. CS
   B. Bubble Sort
   C. Agile
   D. Brogramming

   Answer: C

2. An MVP can be created by

   A. defining the minimum requirements to prove that the product is working as hypothesized.
   B. taking existing code and refactoring it so it is smaller and more compact
   C. minimizing the number of people working on the project so that the knowledge is with one person only

D. coming up with every possible roadblock and feature that might occur and map it out on a Gantt chart

Answer: A

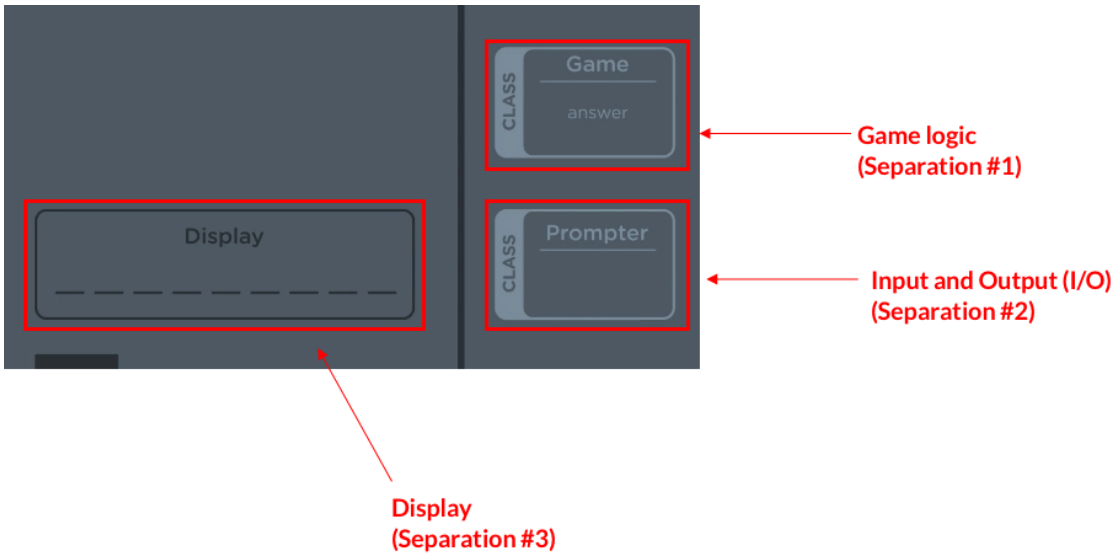
3. An MVP in our context stands for

A. Mostly Verbatim Process
B. Most Valuable Player
C. Minimum Viable Product

Answer: C

3 Getting Started

- Separation of concerns heavily considered
  - The Goal is to make the code as reusable as possible, i.e. The same game logic should be applicable in desktop, website, phone in medium other than console




```
1    public class Game {
2        private String answer;
3
4        public Game(String answer) {
5            this.answer = answer;
```

```
6           }
7       }
8
```

Listing 1: lesson_03/Game.java

```
1       public class Hangman {
2
3           public static void main(string[] args) {
4               Game game = new Game("treehouse");
5           }
6
7       }
8
```

Listing 2: lesson_03/Hangman.java

4 Quiz 1

1. What concerns have we chosen to separate?

   A. We are separating the answer from the guesser.

   B. We have chosen to separate the display of the game from the logic, or state, of the game.

   C. We are separating letters from each other so that we can make spaces.

   Answer: B

2. Due to the separation we have chosen, what outcome can we expect?

   A. We will be able to run this in other languages like JavaScript or Python.

   B. We can more easily generate code using external tools.

   C. We will be able to use the same game logic in other applications, such as console applications, web sites and apps.

   Answer: C

5 Storing Guesses

- *STRING_VAR.indexOf(VAL)*: tells the index of beginning value of *VAL* in string

```
 public class Game {
     private String answer;
     private String hits;
     private String misses;

     public Game(String answer) {
         this.answer = answer;
         this.hits = "";
         this.misses = "";
     }

     public boolean applyGuess(char letter) {
         boolean isHit = answer.indexOf(letter) != -1; // <- this guy here :)
         if (isHit) {
             hits += letter;
         } else {
             misses += letter;
         }

         return isHit;
     }
 }

```

Listing 3: lesson_05/Game.java

6 Exercise 1

- Solution included in *exercise_1.java*

7 Prompting for Guesses

- *Scanner*
  - Is similar to *stdin* in *C*
  - Lives in *java.util* package

```
1   import java.util.Scanner;
2
3   public class Prompter {
4       ...
5
6       public boolean promptForGuess() {
7           Scanner scanner = new Scanner(System.in); // <- This
    little guy here :)
8           ...
9       }
10  }
11
```

Listing 4: lesson_07/Prompter.java

8 Exercise 2

- Solution included in *exercise_2.java*

9 Current Progress

- *STRING_VAR.toCharArray():* Turns string into a char array
  - Is very similar to python's *[x for x in STRING_VAR]*

```
        public class Game {
            public String getCurrentProgress() {
                String progress = "";

                for (char letter : this.answer.toCharArray()) {  // 1.
    <- This little guy here :)
                    ...
                }

                return progress;
            }
        }

```

Listing 5: lesson_09/Game.java

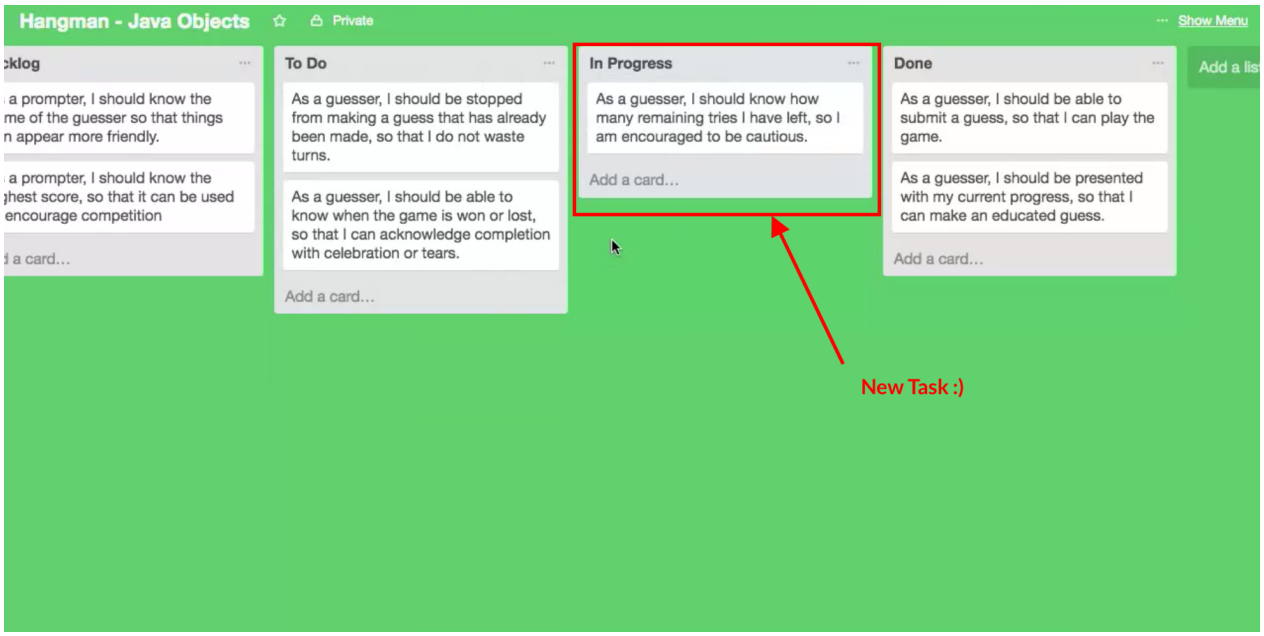
Notes:

- Files can be compiled and displayed by typing *javac Example.java && java Example* in terminal

10 Exercise 3

- Solution included in *exercise_3.java*

11 Remaining Tries



- *STRING_VAR.length()*: Returns the length of string
  - Is similar to *len* in Python

```
1   public class Game {
2       ...
3       public int getRemainingTries() {
4           return MAX_MISSES - this.misses.length(); // <- This little
    guy here :)
5       }
6       ...
7   }
8
```

Listing 6: lesson_11/Game.java

12 Exercise 4

- Solution included in *exercise_4.java*