development log

## checking user answers – 20/12/22

* Mainactivity passes a map <String, Int> to rvadapter, where the string is pizza topping; int is 0-3, where the ints represent the following:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 2 | 3 |
| Not present, user hasn’t selected (color remains unchanged) | Present, user hasn’t selected (yellow) | Not present, user selected (red) | Present, user selected (green) |

* When a new pizza is requested, mainactivity passes a map to rvadapter which has all Int = 0 so the background colors of all the topping items displayed in the rv is neutral.
* As user clicks on toppings to select / deselect them, mainactivity keeps track of a map of <String, Boolean>, where String is a topping; Boolean is whether user has selected it.
* When user clicks “check answers”, mainactivity checks above map against a map of correct answers and produces finalMap<String, Int> which is passed into rvadapter. finalMap has Int from 0-3 as defined in the above table.

## user scores – 24/12/22

|  |  |
| --- | --- |
| **The idea** | When user starts quiz, the app arranges all the pizzas in a random order. When user clicks “next pizza”, the app doesn’t generate a “random” pizza but rather gives the next pizza in the pre-randomised list. There’ll be a score at the top right of the screen which shows how many pizzas the user has gotten right.  A highscore feature could be added in future. |

## solutions fragment – 25/12/22

|  |  |
| --- | --- |
| **The idea** | RecyclerView with all 18 standard pizzas. User can click on a pizza to reveal its toppings, which will be shown in a LinearLayout within the rv\_item. This LinearLayout will be split into 3 columns – “Meats”, “Veg”, “Others” (*Others* will include herbs, garlic, sauces etc.) |
| **Issues / Solutions** | Views get ‘recycled’ and toppings get mixed up and/or duplicated. To fix this, I implemented the stuff in my Notes which makes each view unique and non-recyclable. |

## issue: quiz rv items revert to normal color when user scrolls – 25/12/22

|  |  |
| --- | --- |
| **Planned solution** | I plan to add another variable to the list that Fragment passes to Adapter. This variable (probably a Boolean) tells the adapter whether or not that item has been selected by the user. If it is selected, bg color is changed to light blue, otherwise bg color remains the standard cardview color. |
| **Actual solution** | I used rvMapOfToppings to update rvAdapter on the status of each item – ie selected or unselected by the user as shown below:  rvAdapter.setOnItemClickListener(object : RVAdapterQuiz.onItemClickListener {  override fun onItemClick(position: Int) {  val topping = listOfToppings[position]  if (userMapOfToppings[topping] == true) { // if currently selected  userMapOfToppings[topping] = false // set to unselected  rvMapOfToppings[topping] = 0   } else { // if currently not selected  userMapOfToppings[topping] = true // set to selected  rvMapOfToppings[topping] = 4  }  rvAdapter.notifyDataSetChanged()  } }) |

## highscore feature – 26/12/22

|  |  |
| --- | --- |
| **The idea** | ViewModel reads from json file a variable highScore: Int. highScore is updated when the user plays the quiz and finishes all pizzas if their newScore > highScore. |
| **Issues / Solutions** | Seems like I’m having issues reading a json file that contains just an Int. I’ll try to save a list of Data<String, Int> objects instead, where <String> is the data name, and <Int> is the corresponding value, eg Data(“highScore”, 12). This Data class could also be used for future saved objects.  Instead of the above, I tried saving as a Map<String, Int> but it threw this error when I try reading the json file:  *com.beust.klaxon.Klaxon Exception: Expected a [ but read {*  I’m now trying, instead, to convert each key-value pair into a Data<String, Int>, where Data is a *parcelable* object, then I’ll save a list of Data as a json object. |