Durée du CC : 1h30

Prénom ..... Nom : .....

Le sujet est divisé en deux parties pouvant être traitées de manière indépendante.

1 JSON

Dans les questions suivantes, on suppose que le contenu de `cave.json` (voir la dernière feuille du sujet) est représenté par le `JSONArray array`.

Question 1 Quel est le résultat de `array.getString(0)` ?

Question 2 Quel est le résultat de `array.getJSONObject(2).getInt("millesime")` ?

Question 3 Quel est le résultat de `array.getJSONObject(3).getString(“aoc”)` ?

Question 4 Quel est le résultat de `array.getJSONArray(1).getBoolean(“rouge”)` ?

Question 5 Indiquez la mémoire occupée par la représentation JSON d'un tableau de booléens de taille 8000. Une erreur de quelques octets sera acceptée.

Question 6 Indiquez la mémoire occupée par ce même tableau si on décidait de le représenter d'une manière plus concise.

Question 7 Quelle leçon peut-on tirer des deux questions précédentes ?

2 Application

Dans cette deuxième partie, nous allons écrire une application qui fait une requête via la bibliothèque `volley` sur le fichier `cave.json`, puis qui affiche les différents vins de la cave sous forme d'une liste gérée via une `RecyclerView`.

Certaines lignes de code (notamment les `import`) ont été omises pour simplifier la lecture. Il n'est pas demandé de compléter le code présenté sous forme de `[...]`.

Question 8 *Complétez la méthode `onCreate()` (ligne 22) pour faire en sorte que la requête à l'url `https://u-pec.fr/android/cave.json` soit effectivement émise.*

Question 9 *Complétez le constructeur de `Vin` (ligne 32). On considère dans cet énoncé qu'il n'existe que deux couleurs pour un vin : rouge, ou blanc. Ces couleurs sont représentées par l'enum `Couleur`.*

Question 10 *Complétez le code de la méthode statique `jsonToVin()` (ligne 37).*

Question 11 *Complétez le code de la méthode `setVin()` (ligne 51). Cette méthode a deux objectifs :*

- *remplir les trois `TextView` `aocView`,`domaineView` et `millesimeView` avec les bonnes valeurs, et*
- *colorer le background de la VinView de la couleur du vin, c'est-à-dire en `Color.RED` si c'est un vin rouge ou en `Color.YELLOW` si c'est un vin blanc.*

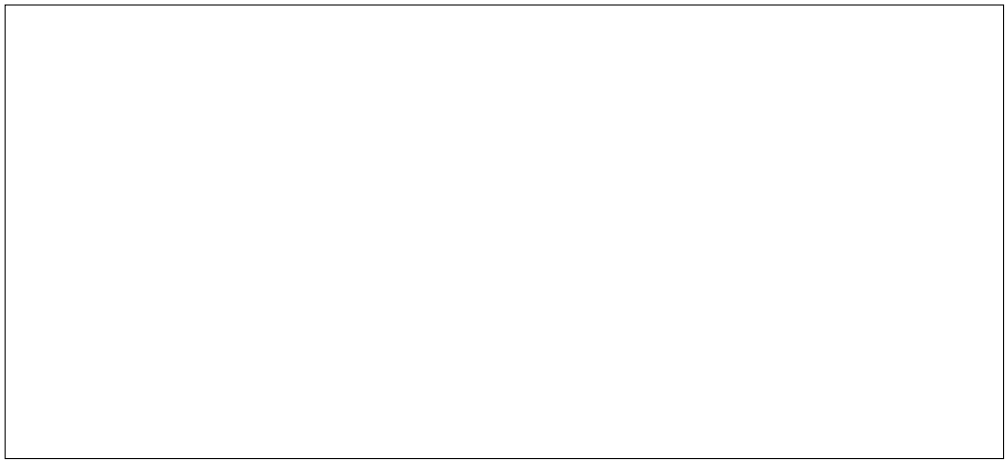
Question 12 *Complétez le code de la méthode `getItemCount()` (ligne 83).*

Question 13 *Complétez le code de la méthode `onBindViewHolder()` (ligne 79).*

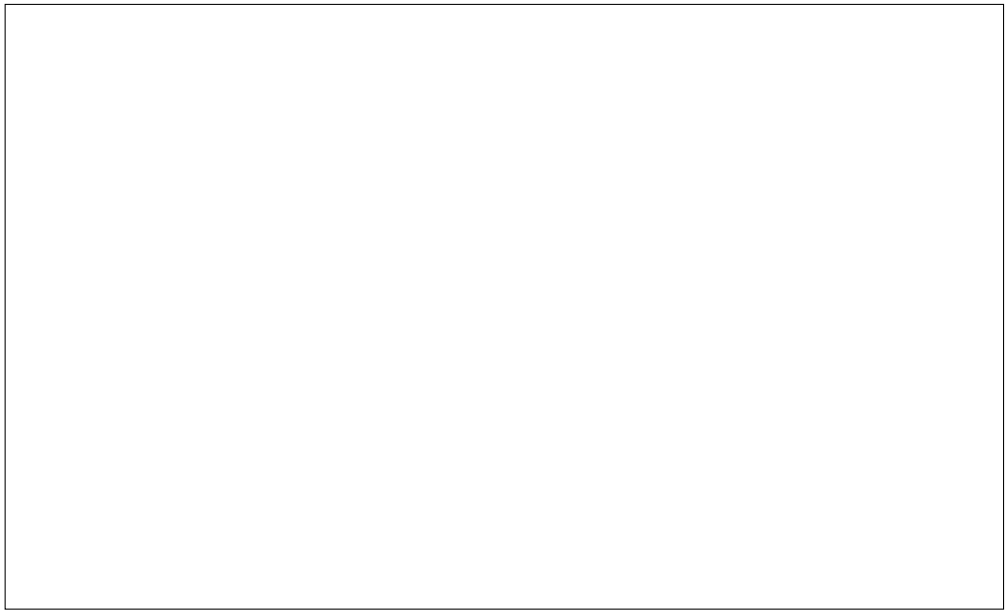
Question 14 *Complétez la déclaration du `Response.Listener<JSONArray>` `lis` (ligne 5). On fera attention à gérer les exceptions : si un vin est mal spécifié dans le JSON, on l'ignorera et on poursuivra le travail sur le reste de la cave.*

```
1  public class MainActivity extends AppCompatActivity {
2    private final static String url = "https://u-pec.fr/android/cave.json";
3    private VinAdapter adapter;
4    private RecyclerView recyclerView;
5    Response.Listener<JSONArray> lis =
6
7    ;
8    Response.ErrorListener errLis = [...];
9
10   protected void onCreate(Bundle savedInstanceState) {
11     super.onCreate(savedInstanceState);
12     setContentView(R.layout.activity_main);
13
14     adapter = new VinAdapter();
15
16     recyclerView = findViewById(R.id.recyclerView);
17     recyclerView.setAdapter(adapter);
18     recyclerView.setLayoutManager([...]);
19
20     RequestQueue queue = Volley.newRequestQueue(this);
21     JsonArrayRequest req = new JsonArrayRequest(url, lis, errLis);
22
23
24   }
25 }
```

```
26  public class Vin {
27
28    private String aoc, domaine;
29    private int millesime;
30    private Couleur couleur;
31
32    public Vin(String aoc, boolean rouge, String domaine, int millesime) {
```



```
33
34    }
35
36    public static Vin jsonToVin(JSONObject jsonVin) throws JSONException {
37      return new Vin(
```



```
38
39      );
40    }
41  }
```

```
42  public class VinView extends ConstraintLayout {
43
44    private TextView aocView, domaineView, millesimeView;
45
46    public VinView(Context context) {
47      [...]
48      // instanciation des TextViews, des contraintes, etc.
49    }
50
51    public void setVin(Vin vin) {
52
53    }
54  }
55
56  public class VinViewHolder extends RecyclerView.ViewHolder {
57
58    private VinView vinView;
59
60    public VinViewHolder(View view) {
61      super(view);
62      vinView = (VinView) view;
63    }
64
65    public void setVin(Vin vin) {
66      vinView.setVin(vin);
67    }
68
69  }
```

```
70  public class VinAdapter extends RecyclerView.Adapter<VinViewHolder> {
71
72    ArrayList<Vin> vins = new ArrayList<>();
73
74    public VinViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
75      VinView vinView = new VinView(parent.getContext());
76      return new VinViewHolder(vinView);
77    }
78
79    public void onBindViewHolder(VinViewHolder holder, int position) {
80
81    }
82
83    public int getItemCount() {
84
85    }
86
87    public void addVin(Vin vin){
88      vins.add(vin);
89    }
90
91  }
92
93  public enum Couleur {
94    ROUGE,
95    BLANC;
96  }
```

Documentation

```
// org.json

class JSONArray {
  boolean getBoolean(int index);
  int getInt(int index);
  String getString(int index);
  JSONArray getJSONArray(int index);
  JSONObject getJSONObject(int index);
  int length();
}

class JSONObject {
  boolean getBoolean(String name);
  int getInt(String name);
  String getString(String name);
  JSONArray getJSONArray(String name);
  JSONObject getJSONObject(String name);
}

class JSONException extends Exception;

// views

class View {
  void setBackgroundColor(int color);
}

class TextView {
  void setText(String str);
}

abstract class RecyclerView.Adapter<T extends RecyclerView.ViewHolder> {
  abstract T onCreateViewHolder(ViewGroup parent, int viewType);
  abstract void onBindViewHolder(T holder, int position);
  abstract int getItemCount();
  void notifyItemChanged(int position);
  void notifyDataSetChanged();
}
```

```
// volley

class RequestQueue {
  void add(Request);
}

class JsonArrayRequest extends Request;

interface Response.Listener<T> {
  void onResponse(T response);
}

// divers

class ArrayList<T> {
  boolean add(T elem);
  T get(int index);
  int size();
}

class Color {
  static int RED;
  static int YELLOW;
}
```

cave.json

Voici le début du fichier JSON `cave.json`, accessible à l'url `u-pec.fr/android/cave.json`

```
[
  {
    "aoc":"Bonnes-Mares",
    "rouge":true,
    "domaine":"Castagnier",
    "millesime":2018
  },
  {
    "aoc":"Pauillac",
    "rouge":true,
    "domaine":"Pichon-Longueville Baron",
    "millesime":2005
  },
  {
    "aoc":"Sauternes",
    "rouge":false,
    "domaine":"Yquem",
    "millesime":1998
  },
  {
    "aoc":"Echezeaux",
    "rouge":true,
    "domaine":"Jean Grivot",
    "millesime":1995
  },
  ...
]
```

Ce fichier représente une cave composée d'un certain nombre de bouteilles de vin. Il n'est pas nécessaire de comprendre cela pour traiter le sujet, mais les différents champs correspondent aux informations suivantes :

aoc : "appellation d'origine contrôlée". Elle correspond à l'origine géographique du vin. Il peut par exemple s'agir de larges régions (`bordeaux`, `bourgogne`) ou de lieux plus précis (un village, ou même une parcelle précise de ce village).

rouge : un booléen décrétant s'il s'agit d'un vin rouge. Pour simplifier, on va considérer qu'un vin est soit rouge, soit blanc.

domaine : le vigneron, ou le château, qui entretient les vignes et vinifie le raisin.

millesime : le millésime correspond à l'année de production d'un vin.