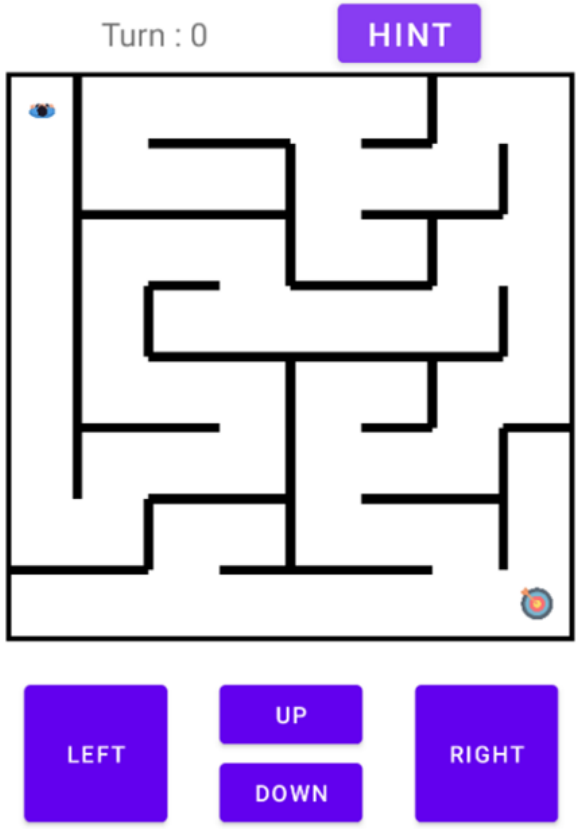
PA2
The Maze Runner

Mobile App Programming

The Maze Runner

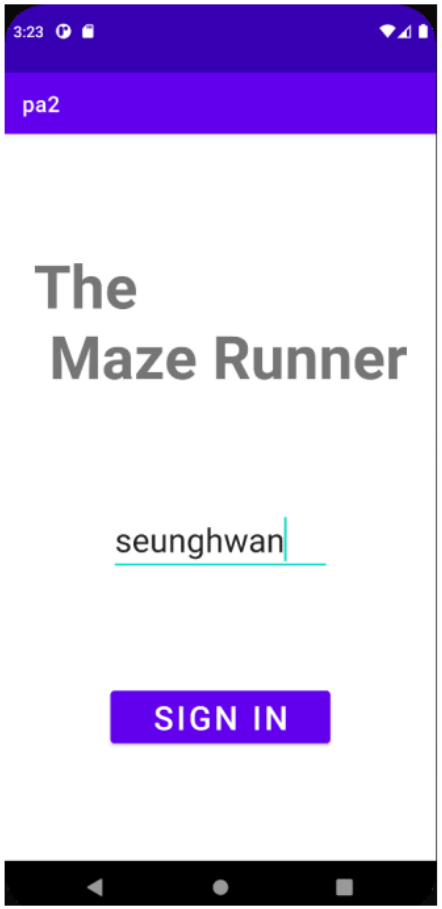
- Escape the maze using Arrow keys!
- There’re 3 activities.
  - SignInActivity
  - MazeSelectionActivity
  - MazeActivity
- You must use ListView, GridView.
  - GridView may not be handled in previous lectures. So please refer to here.
- Your application also contains HTTP networking and explicit Intent.



Activity Explanation – SignInActivty

- This activity check input [**user name**] is valid.
- At first, when **SIGN IN button is clicked**, your application sends a **HTTP POST request** to the server.
  - URL : http://swui.skku.edu:1399/users
  - JSON body : {"username" : [***INPUT_FROM_EDITTEXT***]}
- When the server receives your request, it sends response with following JSON data:
  - {"success": true} if received username exists in below list:

    ["seunghwan", "seongho", "seongmin", "mukoe"]
  - Or your server sends {"success" : false}
- If your application receives **true**, go to **map selection activity**.
- If it receives false, show below toast message **shortly**.
  - "Wrong User Name"



Layout Description – activity_ signin.xml

- Refer to beside image to set constraints
- Title **TextView**
  - Text size : 55sp
  - Text style : bold
  - Width/Height : wrap_content
- Username **EditText**
  - inputType : text
  - Text size : 30sp
  - Width : 200dp
  - Height : wrap_content
- Signin **Button**
  - Text size : 30sp
  - Width : 200dp
  - Height : wrap_content

Activity Explanation – MazeSelectionActivity

- This activity shows the possible maze list.
- The **topmost TextView** prints the username from **"signin" activity**.
- When activity starts, it sends a **HTTP GET request** to the server to retrieve a list of mazes.
  - URL : http://swui.skku.edu:1399/maps
  - No query parameters (body data)
- For this GET request, the server sends back response with following JSON list data: (**JsonArray**)
  - **Ex) [{ "name": "maze1", "size": 5 },**
    { "name": "maze2", "size": 5 },
    { "name": "maze3", "size": 5 },
    { "name": "maze4", "size": 8 },
    { "name": "maze5", "size": 6 },
    { "name": "maze6", "size": 7 }]
- To handle above Json format, refer to **here**!

Activity Explanation – MazeSelectionActivity

- You must print all received mazes using ListView, with ***maze_entry.xml***
- **Ex) [{ "name": "maze1", "size": 5 },**
  { "name": "maze2", "size": 5 },
  { "name": "maze3", "size": 5 },
  { "name": "maze4", "size": 8 },
  { "name": "maze5", "size": 6 },
  { "name": "maze6", "size": 7 }]
- Each Json Object in received JsonArray indicates each line of ListView.
- If user clicks the **START button**, your application opens **MazeActivity**.

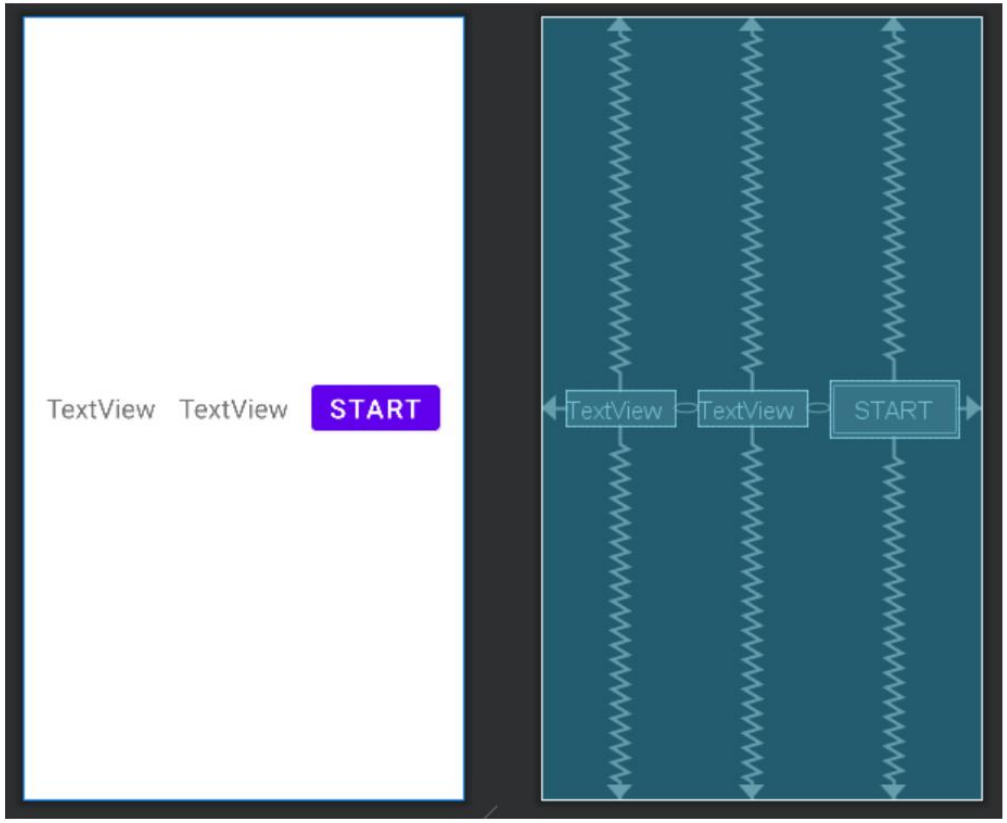
- HINT) The **length** of received JSON list **can be changed** when we test your PA.

Layout Description – activity_ mazeselection.xml

- Username TextView
  - TextSize : “30sp”
  - Width/Height : wrap_content
- Explanation TextView
  - TextSize = “25sp”
  - Width/Height : wrap_content
- MazeList ListView
  - Width : 350dp
  - Height : 500dp

Layout Description – maze_entry.xml

- Mazename TextView
  - TextSize : "25sp"
  - Width/Height : wrap_content
- Mazesize TextView
  - TextSize = "25sp"
  - Width/Height : wrap_content
- Mazestart Button
  - TextSize = "25sp"
  - Width/Height : wrap_content



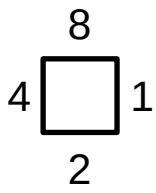
Activity Explanation – MazeActivity

- This activity **gets maze from the server** and shows the selected **SQUARE maze**.
- The number of columns and rows **varies from 5 to 10**.
- GridView's width and height **always set as 350dp**,
  so according to the number of columns and rows, the size of each cell in GridView must be changed. (***350dp / [# of cols or rows]***)
- When activity starts, it sends a **HTTP GET request** to the server to get the **maze shape**.
  - URL : http://swui.skku.edu:1399/maze/map?name=:name
  - Query Parameter**(:name)** : "the name of maze"
    (e.g. "maze1", "maze2", etc.) → Retrieved from MazeSelectionActivity.

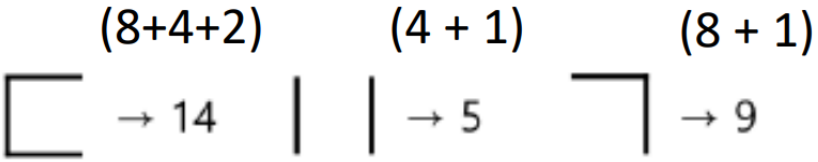
- As a maze shape, the server will send follow Json data (as ***String*** type)
  - EX) { "maze": "5 \n
    14 10 10 9 13 \n
    12 8 9 6 1 \n
    5 5 4 11 5 \n
    5 5 7 12 3 \n
    7 6 10 2 11 \n"}

Activity Explanation – MazeActivity

- You can parse received maze shape data using following rules:
  - The **first line** means **the size**
    (the number of rows and columns)
  - From the second line, there're **(size) * (size)** number of integers.
  - Each integer means an **one maze cell**! (in GridView)
  - **Top** wall is **8**, **Left** wall is **4**, **bottom** wall is **2**, **right** wall is **1**.

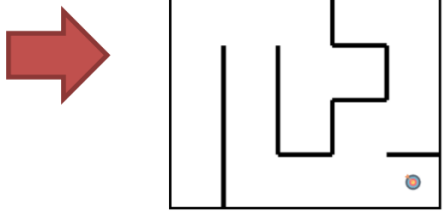


  - So an integer **14** indicates that
    a cell contains **top, left and bottom walls. (14 = 8 + 4 +2)**



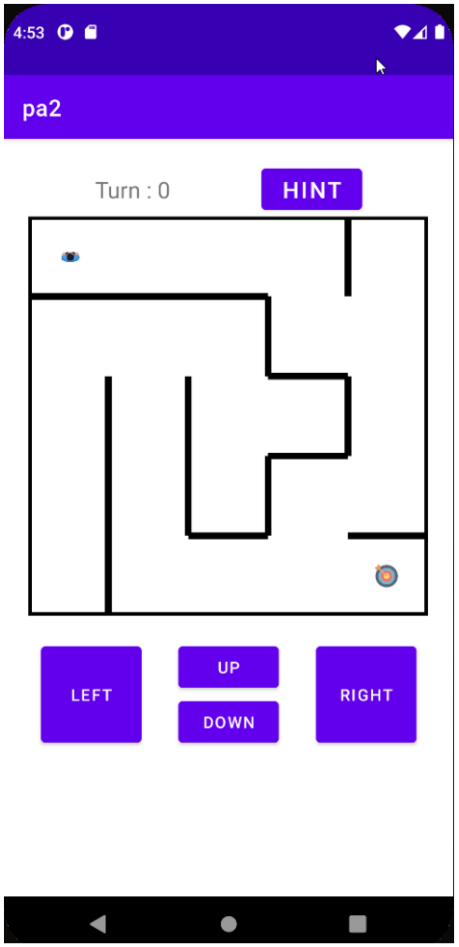
e.g.

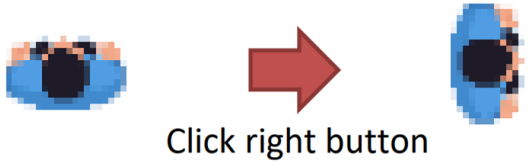
```
5
14 10 10 9 13
12 8 9 6 1
5 5 4 11 5
5 5 7 12 3
7 6 10 2 11
```



Activity Explanation – MazeActivity

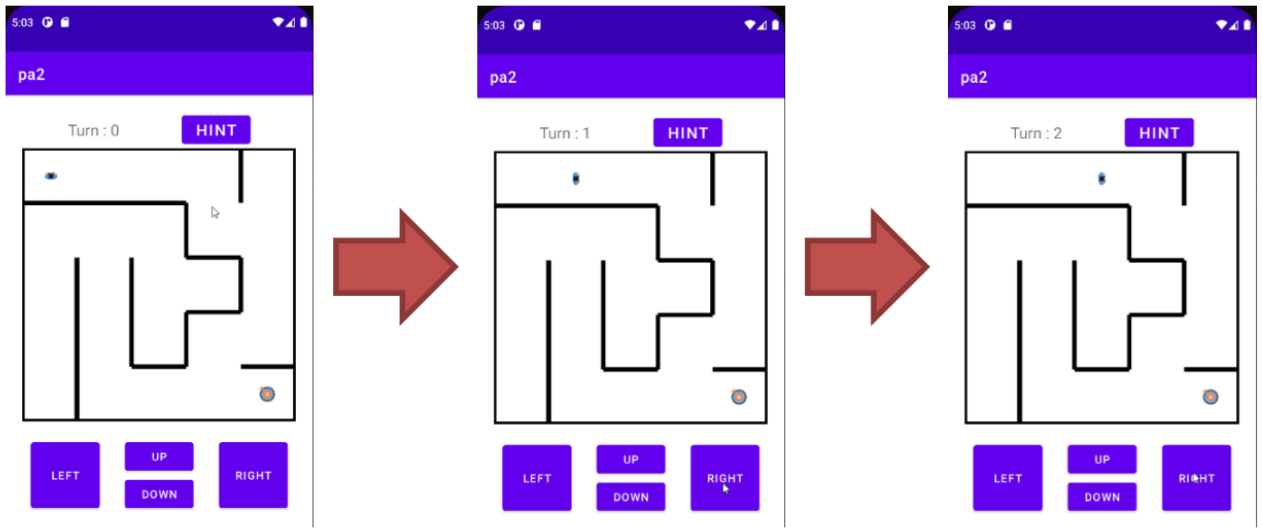
- About the maze runner playing,
  - The start point of the game is always **(0,0)** of GridView.
    - User character must be located on **(0,0) at first**.
  - The goal point is always **(size, size)** of GridView.
    - Goal point is also printed on that grid.
  - If you click **(up, down, left, right)** buttons below, user character will move to following direction.
    - But, user character **CANNOT** pass through the wall.
  - After a certain button is clicked, the user character must also be **moved to that direction**.
    (AND the user character image **must also be rotated in the corresponding direction**)





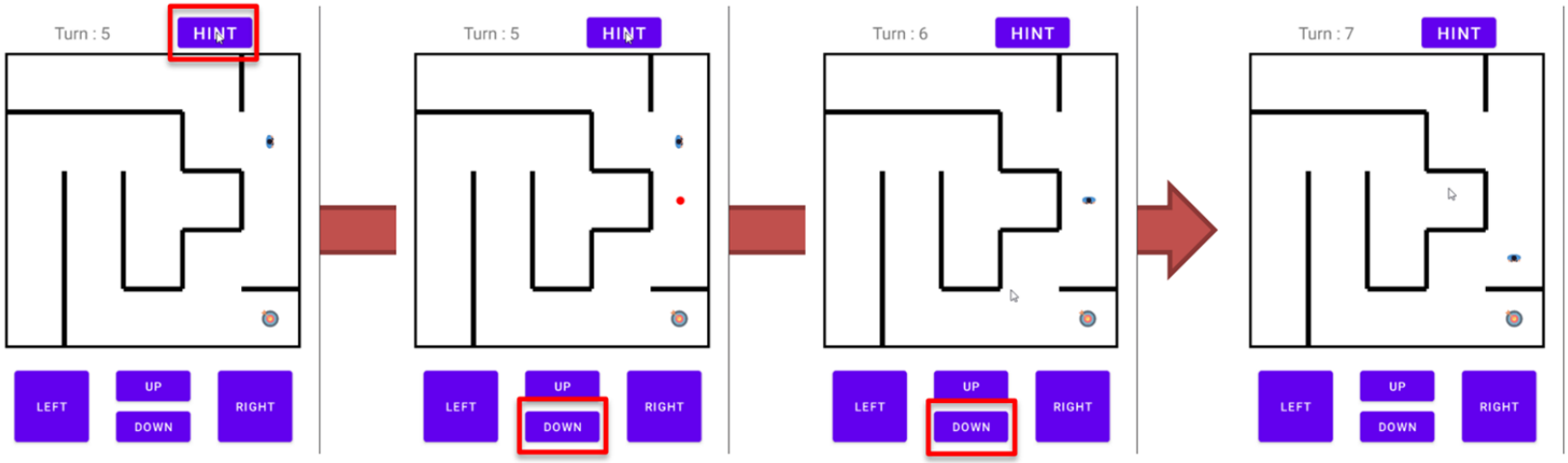
Activity Explanation – MazeActivity

- At the top of the layer, there're "Turn" TextView and "Hint" Button.
  - Turn TextView.
    - It prints "Turn : [# of moves]".
      On every movement (one of the direction button is clicked), increases it.
    - **DO NOT** increase [#of moves] when the character can't move.
      (**Due to the wall**)



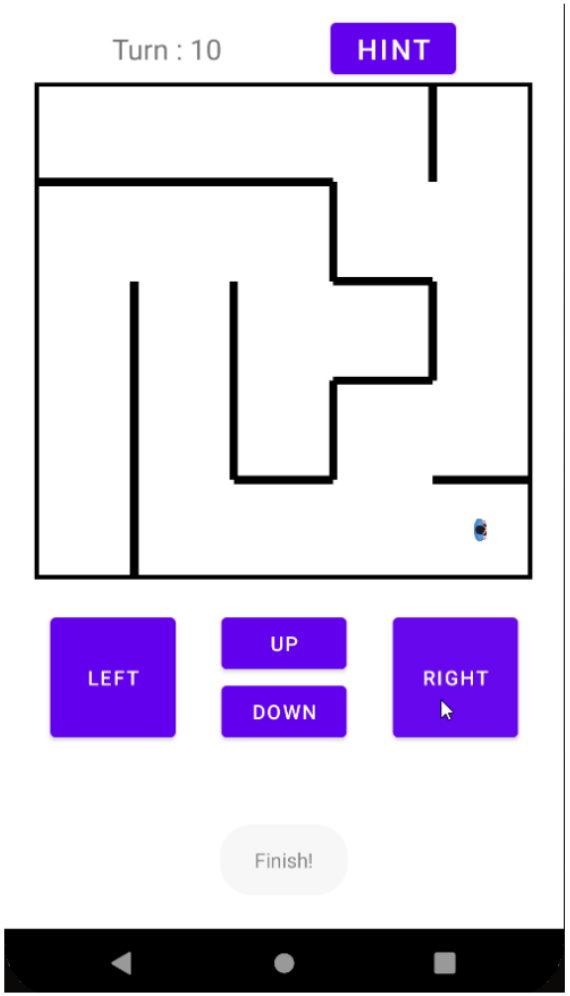
Activity Explanation – MazeActivity

- At the top of the layer, there're "Turn" TextView and "Hint" Button.
  - Hint Button.
    - If user clicks the hint button, your application **calculates the shortest path to the goal point, (size, size)**, and it **draws a dot** on the next place to move among that path.
    - User **ONLY uses hint button once** for each maze.



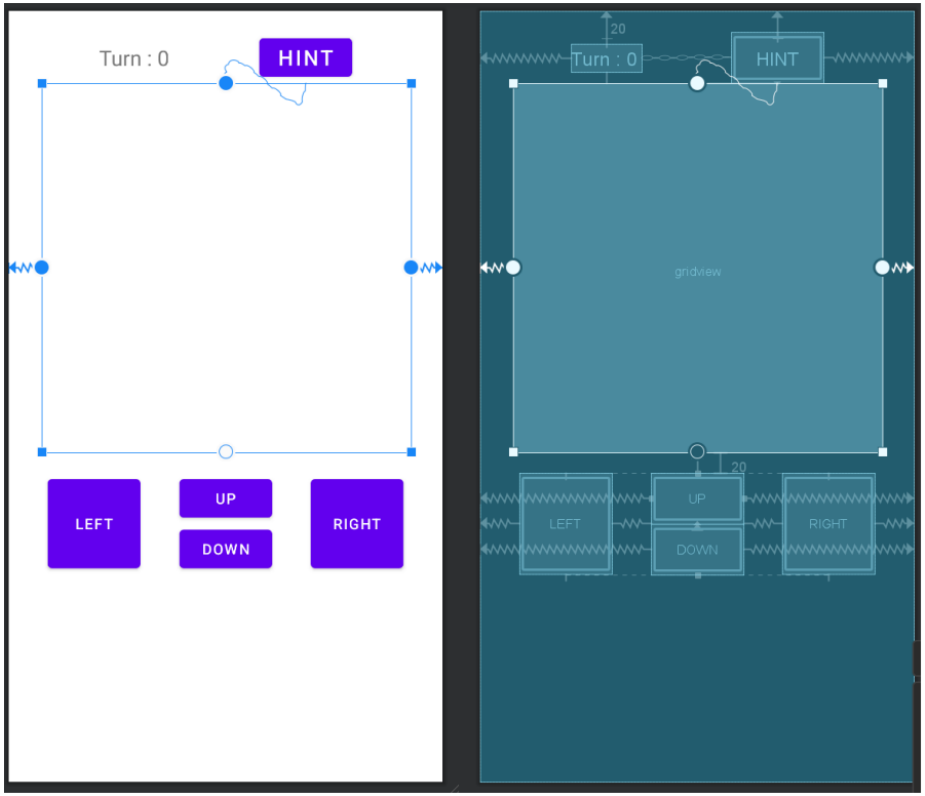
Activity Explanation – MazeActivity

- When the user arrives at the goal point,
  - Just show a toast message shortly, with the **"Finish!"** text.



Layout Description – activity_ maze.xml

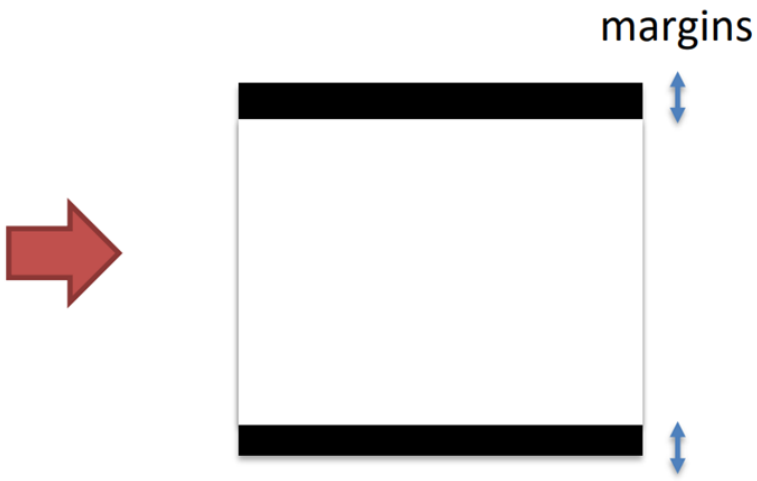
- Turn TextView and Hint Button
  - TextSize : "20sp"
  - Width/Height : wrap_content
- GridView
  - Width/Height : 350dp
- Arrow (Left, Right, Up, Down) Buttons
  - Margin Top : 20dp



Layout Description – maze_cell.xml

- **Recommendation guides** about how to represent maze
  - Students must print maze clearly
  - We recommend you to use Gridview (Very similar with Listview)
    - https://developer.android.com/reference/android/widget/GridView
  - When you generate a cell(item), put two ImageViews in layout file.

```
<constraint layout
      android:background="@color/black" >
<image view
      android:background="@color/white"
      marginTop="3dp"
      marginBottom = "3dp"  />
<image view
      layout_width = "30dp"
      layout_height = "30dp"
/>
</constraint layout>
```



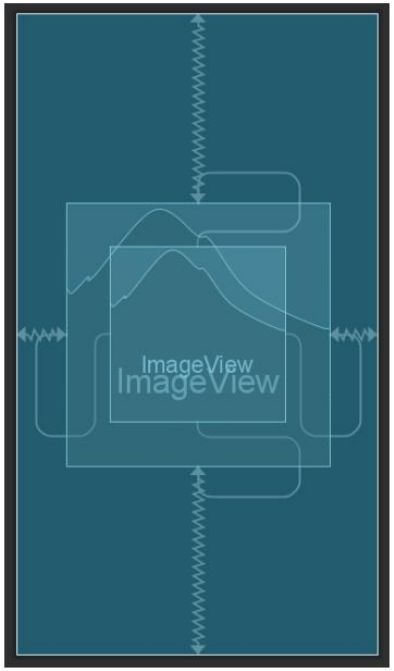
Layout Description – maze_cell.xml

- Therefore, you must set each cell's width, height and margin attributes **in your MazeActivity.kt code programmatically!**

(HINTS)

- At this time, you must consider the dpi, because those attributes are only set in pixels in Kotlin code!

- To put image over the cell (user character, goal and hint dot), make a new ImageView. Next, set layout with following ways:
  - Bottom to bottom of {cell image view}
  - Top to Top of {cell image view}
  - End to End of {cell image view}
  - Start to Start of {cell image view}
- User Character, goal and hint dot's image size is both **30dp**!



Networking - summaries

- TA runs a server program on cloud to handle your HTTP requests.
  - Do not send too many requests. (**Your IP can be banned.**)

- Send to http://swui.skku.edu:1399/

- If the server doesn't work, leaves a question on Google Q&A Sheet.

https://docs.google.com/spreadsheets/d/1GU5vTJjO015x4Cx8NkRDTzkwwm7Z-L6FlTKJbvDTAzE/edit?usp=sharing

- We will upload the server code (python codes) and explain how you run it, so if the server doesn't work, ***please make local server on your computer and use it!***

PA2 Specification

- Project Settings
  - Minimum SDK: Must be 29 (Android 10.0)
  - Target & Compile SDK: Must be 33 (Android 13.0)
  - Application ID(Package): Must be edu.skku.cs.pa2

- Application Execution
  - The application must be started in 10 seconds.
  - UI must not stop more than 5 seconds.
  - No error while build or executions.

PA2 Criteria

- If you application works exactly same with the sample we show you, you can get full points!

- SignInActivity – 20pts
- MazeSelectionActivity – 30pts
- MazeActivity – 50pts

- We deduct your score as **5pts per each error**

PA2 Criteria

- **No restriction on,**
  - # of Adapters, # of classes, # of files

- Use **Google Q&A sheet** for questions!

PA2 Submission

- Submit single zip file with name "<Student ID>_pa2.zip"
  - Shift Twice -> search "export" -> Export to zip -> Change file name and select location to save
  - Do not care about ending '-<Number>' (ex: 2023524288-1.zip)
- Submission Due
  - 5/19 23:59
  - Delayed Submission
    - ~5/21 23:59
    - We will never receive the assignment from May 22
    - Your score will be penalized by 25%p per day.
      - 70/100, 2 day late = 70*(1-0.25*2) = 35/100