Written Re-examination

Programmering af mobile applikationer
Medialogy, 2nd semester

Tuesday 13 August 2024, 9.00 – 13.00

Navn: _____

Studienr: _____

Programmering af mobile applikationer

Re-examination

13 August 2024, 9.00 – 13.00

Instructions

- You have 4 hours to complete this examination.
- Neither electronic devices nor written material are allowed in the examination room.
- This examination consists of 8 questions. Each question is worth 10 marks. You must obtain at least 40 marks to pass.
- Do not write any answers on this question paper—answers written on the question paper will be ignored by the examiner. Write all your answers on the writing paper provided.
- Do not write your answers in pencil and do not use a pen with red or green ink. Use a pen with blue or black ink.
- Hand in no more than one answer to each question.
- Do not turn over until you are told to do so by the invigilator.

Question 1

a) Rewrite the following code using trailing lambda syntax:

```
Row(
    content = {
        Text("Some text")
        Text("Some more text")
        Text("Last text")
    }
)
```

[4 marks]

b) The following code causes a UI component to be previewed in the Preview pane in Android Studio. Draw a sketch of what this UI component will look like.

```
@Composable
fun GreetingText(to: String,
                 from: String,
                 modifier: Modifier = Modifier) {
    Row { this: RowScope
        Text(
            text = "Happy Birthday, $to!",
            fontSize = 30.sp,
            lineHeight = 36.sp
        )
        Text(
            text = "From $from",
            fontSize = 24.sp)
    }
}

@Preview(showBackground = true)
@Composable
fun BirthdayCardPreview() {
    HappyBirthdayTheme {
        GreetingText(
            to = "Dave",
            from = "Susanne"
        )
    }
}
```

[6 marks]

Question 2

Study the following code snippet which defines a UI that will be shown in the Preview pane in Android Studio:

```
41  @Composable
42  fun DiceWithButtonAndImage(
43      modifier: Modifier = Modifier
44  ) {
45      Column (
46          modifier = modifier,
47          horizontalAlignment = Alignment.CenterHorizontally
48      ) { this: ColumnScope
49          Image(
50              painter = painterResource(id = R.drawable.dice_1),
51              contentDescription = "1")
52          Button(onClick = {/*TODO*/}) { this: RowScope
53              Text(stringResource(R.string.roll))
54          }
55      }
56  }
57
58  @Preview
59  @Composable
60  fun DiceRollerApp() {
61      DiceWithButtonAndImage(modifier = Modifier
62          .fillMaxSize()
63          .wrapContentSize(Alignment.Center)
64      )
65  }
```

a) When calling the DiceWithButtonAndImage function, is it necessary to provide a modifer argument? Briefly explain your answer. [2 marks]

b) What is the effect of the chained call to fillMaxSize() in line 62? [2 marks]

c) What is the effect of the chained call to wrapContentSize() in line 63? [2 marks]

d) Assume that R.string.roll is the resource id of a String resource whose value is “Roll”. In what file is this string resource defined? [2 marks]

e) Assume that R.drawable.dice_1 is the resource id of the following image:

Draw a sketch of the UI that the code above would cause to be shown in the Preview pane in Android Studio. [2 marks]

Question 3

a) In each of the following situations, state whether it would be more appropriate to use a LazyColumn composable or a Column composable:
   i. A list with a small, fixed number of items to display.
   ii. A list of items to display that requires a scrollbar.
   iii. A possibly long list of items where we don't know beforehand how many items there will be in the list.

   [3 marks]

b) Why should you generally provide more than one bitmap image for an app's icon? [2 marks]
c) What is the name given to a class that can only have one instance? [2 mark]
d) Give an example of a scenario where you would want a class to only have one instance. [2 mark]
e) In Kotlin, how do you define a class that only has one instance? [1 mark]

Question 4

For each of the following, state which Activity lifecycle callback method is invoked when the lifecycle state changes

a) from Started to Resumed;
b) from Created to Destroyed;
c) from Created to Started, when the state before Created was also Started;
d) from Started to Created;
e) from Initialized to Created.

[2 marks for each part]

Question 5

Study the following code snippet and answer the questions that follow it.

```
1  private const val BASE_URL =
2      "https://android-kotlin-fun-mars-server.appspot.com"
3
4  private val retrofit = Retrofit.Builder()
5      .addConverterFactory(ScalarsConverterFactory.create())
6      .baseUrl(BASE_URL)
7      .build()
8
9  interface MarsApiService {
10     @GET("photos")
11     suspend fun getPhotos(): String
12 }
13
14 object MarsApi {
15     val retrofitService : MarsApiService by lazy {
16         retrofit.create(MarsApiService::class.java)
17     }
18 }
```

a) In Retrofit, what is the purpose of a converter factory and in which line in the code above is such a factory created? [3 marks]
b) What is the purpose of line 10? [3 marks]
c) In which lines of code is a singleton defined? [1 marks]
d) What is "lazy initialization"? [2 mark]
e) In the code above, which object is lazily initialized? [1 mark]

Question 6

Study the following code which provides an example of a Room Data Access Object, and answer the questions that follow it.

```
11  @Dao
12  interface ItemDao {
13      @Insert(onConflict = OnConflictStrategy.IGNORE)
14      suspend fun insert(item: Item)
15
16      @Update
17      suspend fun update(item: Item)
18
19      @Delete
20      suspend fun delete(item: Item)
21
22      @Query("SELECT * from items WHERE id = :id")
23      fun getItem(id: Int): Flow<Item>
24
25      @Query("SELECT * from items ORDER BY name ASC")
26      fun getAllItems(): Flow<List<Item>>
27  }
```

a) In Room, what is the purpose, in general, of a Data Access Object? [2 marks]
b) Why are the first three methods in ItemDao above marked "suspend"? [2 marks]
c) If a conflict arises when a new item is inserted into the database, what happens? [2 marks]
d) What are the advantages of having the query methods return objects of type Flow? [2 marks]
e) Why are the final two query methods (lines 22 – 26) *not* defined as suspend functions?[2 marks]

Question 7

a) WorkManager provides two types of worker class, Worker and CoroutineWorker. Suppose you want to have WorkManager perform a task asynchronously on a background thread.
    i. Which of these two worker classes would you extend? [2 marks]
    ii. Which method in your selected base class (superclass) would you override? [2 marks]
b) What class of objects is used to represent the input and output of a worker in WorkManager? [2 mark]
c) What is a WorkRequest object used for in WorkManager? [2 marks]
d) What are the two concrete implementations of WorkRequest that are provided by the WorkManager library? [2 marks]

Question 8

Each of the following statements refers to the topic of building Android apps with Views. For each statement, write down whether it is true or false.

a) Layouts are typically declared using XML.
b) Each component in the UI necessarily contains a Layout.
c) A View is a special type of Layout.
d) A ViewGroup is a special type of View that can contain other Views (which may themselves be ViewGroups).
e) When building an app with Views, a Fragment that hosts the XML layout replaces the concept of a Composable "screen".

[2 marks for each correct part]

END OF EXAMINATION