UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Written Examination

DIT341 – Web and Mobile Development

Tuesday, January 8th, 2018, 14:00 - 18:00

Examiner: Philipp Leitner

Course Responsible: Grischa Liebel

Contact Persons During Exam:
Philipp Leitner (+46733056914)
Joel Scheuner (+46 733 42 41 26)

Allowed Aides:
None except English dictionary (non-electronic), pen/pencil, ruler, and eraser.

Results:
Exam results will be made available no later than 15 working days after the exam date through Ladok.

Overall Points: 50

Grade Limits: 0 – 24 points: **U**, 25 – 42 points: **G**, >42 points: **VG**

Review:
The exam review will take place latest three weeks after the exam results have been published in Ladok. It will be announced on GUL at least one week in advance.

Part 1 – Concept Definitions & Explanations (17P)

Q1.1: Describe and briefly contrast server-side and client-side development. **(2P)**

Q1.2: Describe briefly how HTTP requests and responses are structured, and provide a valid example for a (successful) HTTP exchange. **(5P)**

Q1.3: Name and describe the three layers of a three-tiered Web architecture. **(3P)**

Q1.4: Describe briefly what events in the lifecycle of an HTML web page would trigger the `onload` and `onsubmit` Javascript event handlers. **(2P)**

Q1.5: Define what a single page app (SPA) is, especially in comparison to other Web applications. **(1P)**

Q1.6: Explain the idea of HATEOAS ("hypermedia as the engine of application state"). **(1P)**

Q1.7: Briefly explain the concept of threat modelling (in the context of security of Web and mobile apps). **(1P)**

Q1.8: Describe the standard folder structure of an Android application. Which information / data goes where? **(2P)**

Part 2 – Working with code (19P)

HTML and CSS

Q2.1: Identify and briefly describe 4 errors in the malformed xHTML document shown in Figure 1. You can ignore line 2. **(2P)**

```
<?xhtml version=”1.0” encoding=”UTF-8”?>
<!DOCTYPE html PUBLIC ...>

<html xmlns=”http://www.w3.org/1999/xhtml”>
    <head>
        <title content=”HTML Example 8”/>
    </head>
    <body>
        <h1>Basic HTML Tags (Content)</h1>
        <p><b>Important:</p>
        This message is bold</b>
        <img src=”html5.png” alt=”HTML5 Logo”>
    </body>
</html>
```

Figure 1: A malformed xHTML Document

Q2.2: Figure 2 depicts an HTML web page with a CSS style definition. Describe in which color the text in lines 21-22 and 24-27 is rendered and why. **(3P)**

```html
<!doctype html>
<html>
  <head>
    <style>
      #section1 {
        color: purple;
      }
      p {
        color: orange;
      }
      .skyClass {
        color: blue;
      }
      #grassId {
        color: green;
      }
    </style>
  </head>
  <body>
    <div id="section1">
        <h1>Title1</h1>
        <p class="skyClass">Text2</p>
    </div>
    <p class="sunClass">Text3</p>
    <p>Text4</p>
    <p id=grassId style="color: white;" class="skyClass">Text5</p>
    <p id="grassId" class="skyClass">Text6</p>
  </body>
</html>
```

Figure 2: HTML and CSS

JavaScript

Q2.3: Figure 3 depicts a part of a JavaScript program. Describe what the program outputs in line 2, 4, and 6. **(1.5P)**

```
1  for (var x = 0; x < 3; x++) {}
2  console.log(x);
3  y = 1;
4  console.log(x + y);
5  var y;
6  console.log(y);
```

Figure 3: JavaScript Code

Q2.4: Which JavaScript behaviour is depicted in Figure 3? Name and describe the behaviour. **(1.5P)**

Q2.5: Figure 4 depicts a short JavaScript program. Describe what the console output of the program is and explain why this is the case. **(3P)**

```javascript
function fun1() {
    fun2();
    console.log("1");
}
function fun2() {
    setTimeout(function() {
        console.log("2");
    }, 0);
    console.log("3");
}
function fun3(cb) {
    cb();
    console.log("4");
}

fun3(function() { console.log("5"); });
fun1();
console.log("6")
```

Figure 4: Short JavaScript Program

Frontend Development

Q2.6: Figure 5 depicts a Vue.js program that makes use of a watched property. Describe how a ”watched property” can be used? What alternative Vue construct can be used instead? **(2P)**

Q2.7: Describe the functionality of the Vue.js Application depicted in Figure 5. What is the value of `output` after pushing the ”Click” button? **(2P)**

```
<body>
  <div id="app">
    <h1>test</h1>
    <my-comp></my-comp>
  </div>

  <script>
    Vue.component('my-comp', {
      data: function () {
        return {
          count: 3,
          output: 0
      }},
      template: `<div><button v-on:click="count++">Click</button><br/><p>{{output}}</p></div>`,
      watch: {
        count: function (newNumber, oldNumber) {
          this.output = newNumber + oldNumber;
        }
      }
    });

    var app = new Vue({
      el: '#app'
    });
  </script>
</body>
```

Figure 5: Simple Vue.js Application With Components

Android Development

Q2.8: Figure 6 depicts the use of the Volley library for a potentially long-running operation within Android. What is the advantage of using the Volley library compared to using the built-in `HttpURLConnection`? **(2P)**

Q2.9: What does the following code in Figure 6 do? What is the value of the text view both in case of a) success and b) error? **(2P)**

```java
final TextView textView = (TextView) findViewById(R.id.text);
// ...

RequestQueue queue = Volley.newRequestQueue(this);
String url ="http://www.google.com";

StringRequest stringRequest = new StringRequest(
  Request.Method.GET, url,
  new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        textView.setText(response.reverse());
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        textView.setText("Error!");
    }
});

queue.add(stringRequest);
```

Figure 6: Android Method Performing a Long-Running Operation

Part 3 – REST API Case (8P)

Confluence is a hosted collaboration solution for software developers. The product offers a RESTful API with access to different resources, such as WIKI entries and blog posts. In Figure 7, you see an excerpt of possible HTTP requests for a modified version of the Confluence API.

```
GET /confluence/users
GET /confluence/blogposts/:user
GET /confluence/blogposts/:user/:blogid

POST /confluence/blogposts/:user
PUT /confluence/blogposts/:user/:blogid
POST /confluence/blogposts/:user/:blogid?method=delete
```

Figure 7: Excerpt of Modified Confluence API

Q3.1: Assuming that the API has been designed following the REST principles introduced in this course, describe what content and status code you would expect the following HTTP request to return (assume a successful execution). **(2P)**

- GET /confluence/users

Q3.2: Assuming that the API has been designed following the REST principles introduced in this course, contrast the following two requests. What is your expectation of how their functionality differs? In which cases would you prefer one over the other? **(2P)**

- POST /confluence/blogposts/:user
- PUT /confluence/blogposts/:user/:blogid

Q3.3: Investigate the following operation. What's "wrong" with this API design (i.e., not according to the principles of RESTFul API design), and how should this operation look like instead? **(1P)**

- POST /confluence/blogposts/:user/:blogid?method=delete

Q3.4: Assume that you use the following API operation in your program.

- GET /confluence/blogposts/:user

Name 2 status codes *not* indicating success that your application should be prepared for, and discuss briefly which erroneous situations they would represent. **(2P)**

Q3.5: What do you expect to happen if you send the exact same request *twice* to the following endpoint, immediately after each other? **(1P)**

- PUT /confluence/blogposts/:user/:blogid

Part 4 – Reflection (6P)

Q4.1: In the lecture we have discussed the fundamental properties of RESTFul systems (statelessness, layering, uniform interface, and caching). Reflect on how these properties in combination enable the scalable Internet that we know today. Why is each of these properties important? **(4P)**

Q4.2: Contrast the “smoke testing”, “canary testing”, and “monkey testing” test straegies that we have encountered in the context of testing Android applications. Are those useful only for mobile apps? **(2P)**