UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Written Examination

DIT341 – Web and Mobile Development

Tuesday, January 7th, 2020, 14:00 - 18:00

Examiner: Philipp Leitner

Contact Persons During Exam:
Philipp Leitner (+46 733 05 69 14)
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.

Total Points: 100

Grade Limits: 0 – 49 points: **U**, 50 – 84 points: **G**, >=85 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 Canvas at least one week in advance.

1 Conceptual Questions (30P)

Q1.1: Compare the three ways to declare variables in JavaScript (var, let, const). **(3P)**

Q1.2: Outline the structure of an HTTP request. What mandatory/optional elements can be present in a *request*? **(3P)**

Q1.3: Contrast the usage of the <p>, , and tags in HTML. **(3P)**

Q1.4: What elements does the following combinator in CSS match: `p ~ span`? **(1P)**

Q1.5: What is AJAX? What is the key difference between an AJAX request and other HTTP requests? **(3P)**

Q1.6: Explain the HATEOAS idea of RESTful services. **(2P)**

Q1.7: Explain the idea of two-way data binding in Vue.js. **(2P)**

Q1.8: What is the difference between processes and threads (in Android)? Explain and contrast both. **(2P)**

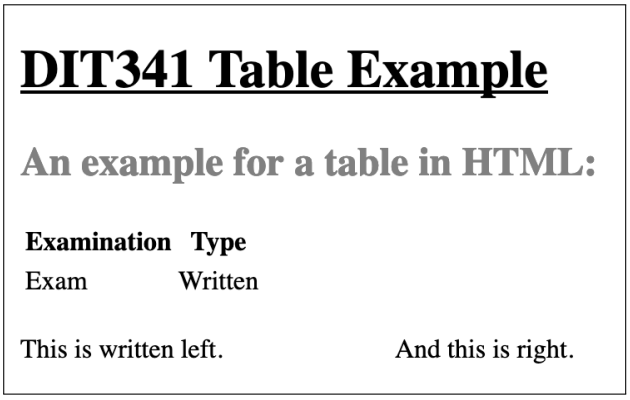
Q1.9: Explain how name resolution in DNS works. Use an example domain name. **(3P)**

Q1.10: Define and explain load and stress testing for Web applications. **(2P)**

Q1.11: Define and explain the three cloud computing service models. Provide an example service for each model. *Hint: X-as-a-Service.* **(6P)**

2 HTML, CSS, and JavaScript (8P)

Complete the HTML code snippet on the next page of the exam paper, so that a browser will display the following view (the border is not part of the HTML document that you need to create). Make use of the styles already defined:



Assume that the default text color of the browser is black. Further, assume that the JavaScript function `looked_at` shall be invoked whenever a user moves the mouse pointer over the table. Write down the missing code on an extra paper and refer to the lines in the template via line numbers. Available space is not necessarily indicative of how much code is required.

```
<!doctype html>
<html>
  <head>
    <style>
      h1 { text-decoration: underline }
      h2 { color: grey }
      th { font-weight: bold; }
      .left { width: 45%; float:left; }
      .right { width: 45%; float:right; }
    </style>
    <script>
        function looked_at() {
          console.log('User has scrolled over the table.')
        }
    </script>
  </head>
  <body>









  </body>
</html>
```

Figure 1: Complete the Blanks (HTML / CSS / JS)

3 Understanding CSS Formatting (14P)

Complete the simple HTML snippet below. Fill out empty spaces so that each line is colored according to the instructions in text (e.g., a line that reads “This line should be entirely in red.” should be rendered in red). Use the predefined CSS styles (you are not allowed to define your own styles, not even inline). Write down the missing code on an extra paper and refer to the lines in the template via line numbers. Available space is not necessarily indicative of how much code is required, and some empty space will not require code. Assume that the default browser color is black.

```
<!doctype html>
<html>
  <head>
    <style>
      div { color: yellow; }
      div > span { color: green; }
      p[custom="pink"] { color: pink; }
      #red { color: red; }
      .blue { color: blue; }
    </style>
  </head>
  <body>
      <div          > This line should be entirely blue.
        <p            >This line should be entirely red.</p>
      </div>
      <p           >This line should be entirely black (even
          within
        <span          >the span</span>).
      </p>
      <div          >This line should be mostly yellow.
        (but the text within <span          >the span</span>
        should be green).
      </div>
      <             >
        <p            >This line should be entirely pink.</p>
      </       >
  </body>
</html>
```

Figure 2: Understanding CSS

4 Frontend Development (14P)

Complete the code snippet of a Vue.js app on the next page of the exam paper. Write an app that initially renders the contents of the array “items” in an unordered list below the header. Further, the app should render a button which, when pressed, appends the content of “to_add” to the displayed list. Pressing the button multiple times will lead to the same content being appended multiple times.
Write down the missing code on an extra paper and refer to the lines in the template via line numbers. You will need to add code in lines 9 – 14 and 26 – 28. Available space is not necessarily indicative of how much code is required. There is no need to interact with a backend in this task.

```
1  <!doctype html>
2  <html>
3  <head>
4      <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js
           "></script>
5  </head>
6  <body>
7    <div id="vueapp">
8      <h1>List of items:</h1>
9
10
11
12
13
14
15   </div>
16
17   <script>
18         var app = new Vue({
19             el: '#vueapp',
20             data: {
21                 items: ["DIT032", "DIT341"],
22                 to_add: "DIT666"
23             },
24             methods: {
25                 fetch: function () {
26
27
28
29                 }
30             }
31         });
32     </script>
33 </body>
34
35 </html>
```

Figure 3: Complete the Blanks (Express)

5 Android Development (9P)

Discuss the following Android code snippet that is using the Volley library to send HTTP requests to a Web backend. Specifically answer the following questions:

- What does the code do (in general)?
- What happens in case of a successful HTTP request (and in case of an unsuccessful) one? Specifically, what happens if only one of the two HTTP requests is successful?
- In what order are the two text views filled with text?

```
final TextView txtViewChalmers =
  (TextView) findViewById(R.id.chalmers);
final TextView txtViewGU =
  (TextView) findViewById(R.id.GU);
// ...

RequestQueue queue = Volley.newRequestQueue(this);
String url1 ="http://www.chalmers.se";
String url2 ="https://www.gu.se";

StringRequest request1 = new StringRequest(
  Request.Method.GET, url1,
  new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        txtViewChalmers.setText(response);
    }
}, null);
StringRequest request2 = new StringRequest(
  Request.Method.GET, url2,
  new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        txtViewGU.setText(response);
    }
}, null);
queue.add(request1);
queue.add(request2);
```

Figure 4: Android Volley Requests

6 REST API Case Study (25P)

Google Docs is a service that allows users to create, edit, and share documents online. Below you see an excerpt from a possible REST-based API for such a service.

```
GET /:user/documents
POST /:user/documents
PATCH /:user/documents/:doc
POST /:user/documents/:doc?delete=true
```

Figure 5: Excerpt of an Imaginary Google Docs API

Answer the following questions about this API, assuming that it has been designed following the REST principles introduced in the lecture.

Q6.1: Describe what functionality and behavior you would expect from the following two API methods. What response would you expect from the service in case of a successful requests? **(4P)**

- GET /:user/documents
- POST /:user/documents

Q6.2: Sketch a legal HTTP request to the API method below, which you would assume to be successful. **(4P)**

- GET /:user/documents

Q6.3: Assume that you use the following API operation in your program. Name two status codes *not* indicating success that your application should be prepared for, and discuss briefly which erroneous situations they would represent. **(4P)**

- PATCH /:user/documents/:doc

Q6.4: Inspect the API method below. Do you consider it good RESTful design? How could it be improved? **(2P)**

- POST /:user/documents/:doc?delete=true

Q6.5: What is the purpose of the following API operation? What would you expect it to do? What parameters would you expect to have to pass to this method? **(4P)**

- PATCH /:user/documents/:doc

Q6.6: Define the concepts of safety and idempotency (for HTTP methods in general). Which of the 4 operations in the above API excerpt would you expect to be safe, and which would you expect to be idempotent? Provide an example of an additional method (following the same patterns as the existing methods) that would be idempotent (but not safe). **(7P)**