Week 3 Graded Questions

Q1: Which of the following are examples of system level state that needs to be maintained for the entire app?

A. User database of NPTEL
B. List of emails selected for archiving in Gmail
C. Google search index
D. User shopping cart content on Amazon.in

Ans: A and C

Explanation: User preference is frontend state, shopping cart is user specific application state.

Q2: Which of the following could be considered user specific application state?

A. User database of NPTEL
B. List of emails selected for archiving in Gmail
C. Google search index
D. User shopping cart content on Amazon.in

Ans: D

Explanation: User shopping cart content is specific to the user. So, it can be considered as user specific application state.

Q3: When you view the online discourse forum, there is a line indicating “last visit” - posts above this have been updated since the last time you visited the forum. What kind of state information is this indicator conveying?

A. Ephemeral UI state
B. Application state
C. System state
D. None of the above

Ans: B

Explanation: Although this is temporary and changes each time the user loads the page, it still needs to be stored at the server to keep it up to date.

Q4: Which of the following methods can be used to ensure that the displayed state and system state are kept in sync at all times?

A. Ajax requests on each UI change
B. Periodic reloading of web-page
C. Vue bindings to update data reactively
D. Pure static pages with all updates rendered from server

Ans: A, D

Explanation: Vue bindings by themselves do not synchronize state, and reloading a page will most likely cause loss of any ephemeral information.

Q5: The M in MVC stores ____.

A. Ephemeral UI state
B. Application state
C. System state
D. None of the above

Ans: C

Explanation: The main idea of persistent storage is that you can reconstruct system state from the stored data

Q6. Consider the following Vue application with markup index.html and JavaScript file app.js,

index.html:

```
<div id="app">{{newMessage}}</div>
```

app.js:

```
const dataObj = {
  message: 'Welcome',
}

const optObject = {
  el: '#app',
  data: dataObj,
}

const app = new Vue(optObject)
app.newMessage = 'Welcome to iitm online degree'
```

What will be rendered by the browser?

A. Welcome
B. Welcome to iitm online degree
C. App with give a warning and will show a blank page.
D. None of the above

Answer: C

Explanation: properties in data are only reactive when they are present when instance was created and “newMessage” is not present in data object it was assigned after the instance was created so, app will not be able to identify the property.

Q7. Consider the following Vue application with markup index.html and JavaScript file app.js,

index.html:

```
<div id="app">{{count+' second remaining'}}</div>
```

app.js:

```
const dataObj = {
  count: 10000,
}

const optObject = {
  el: '#app',
  data: dataObj,
}
```

```
const app = new Vue(optObject)

setInterval(() => {
  app.count -= 1
}, 1000)
```

What will be rendered by the browser after 1 minutes?

A. 10000
B. 9999
C. 9940
D. None of the above

Answer: C

Explanation: After each 1 second callback function of setInterval will reduce the count property of data object by 1 so after 1 minute it well be reduced by 60 so, answer is 9940.

Q8. Consider the following Vue application with markup index.html and JavaScript file app.js,

index.html:

```
<div id="app" style="color: white"
v-bind:style="divStyle">{{message}}</div>
```

app.js:

```
const dataObj = {
  message: 'IITM online',
  divStyle: {
    backgroundColor: 'red',
    padding: '20px',
    fontSize: '2em',
  },
}

const optObject = {
  el: '#app',
  data: dataObj,
}

const app = new Vue(optObject)
```

What will be the color, background color and font-size of the div with ID “app”?

A. White, red, 2em
B. Black, white, 1em
C. white, white, 1 em
D. Black, red, 2em

Answer: A

Explanation: v-bind will bind the style property of div element with divStyle property of data object, and it already has color white assigned so, option A is correct.

Q9. Consider the following Vue application with markup index.html and JavaScript file app.js,

index.html:

```
<div id="app" v-bind:style="{fontSize:fontSize+'em'}">
   <p>{{message}}</p>
   <button v-on:click="zoom">click me</button>
</div>
```

app.js:

```
const dataObj = {
  message: 'IITM online',
  fontSize: 2,
}

const optObject = {
  el: '#app',
  data: dataObj,
  methods: objMethods,
}
```

```
const app = new Vue(optObject)
```

If the font size of the text ‘IITM Online’ increases by 1 em each time the user clicks on the button, which one of the following is the definition of objMethod?

A.
```
const objMethods = {
  zoom() {
    this.fontSize += 1em
  },
}
```

B.
```
const objMethods = {
  zoom() {
    fontSize += 1
  },
}
```

C.
```
const objMethods = {
  zoom() {
    this.fontSize += 1
  },
}
```

D.
```
const objMethods = {
  zoom() {
    fontSize += 1em
  },
}
```

Answer: C

Explanation: Each time user clicks on the button, zoom method will be triggered, and style property is bind with the object {fontSize:fontSize+'em'} so, in order to change the font size we zoom need to change the fontSize property of data object. So, Option C is correct.

Q10. Consider the following Vue application with markup index.html and JavaScript file app.js,

index.html:

```
<div id="app">
     <p directive1>{{message}}</p>
     <button directive2>click</button>
 </div>
```

app.js:

```
const app = new Vue({
  el: '#app',
  data: {
    message: 'Hello',
    seen: true,
  },
  methods: {
    toggleSeen() {
      this.seen = !this.seen
    },
  },
})
```

If you want to toggle between the presence of p element on click of the button element, what are the possible value of directive1 and directive2 and their respective values?

A. v-if=”seen”, v-on:click=”toggleSeen”
B. v-if:”seen”, v-on:click:”toggleSeen”
C. v-if=”seen”, @click=”toggleSeen”
D. v-if:”seen”, @:click:”toggleSeen”

Answer: A and C

Explanation: v-on will bind the click event to the method “toggleSeen” which will change the value of seen property according to the previous set property. And v-if will control the display of p element based on truth value of seen property. And @ is shorthand for v-on. So, A and C are correct.