



SURNAME: PHOENIX

FORENAME(S): SAMMY

Examination Answer Books

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	P151028
2	Title of Examination Paper	EMERGING WEB TECHNOLOGIES
3	Module Code	~~MR~~ M6X01096
4	Date and Time of Examination	09/05/2018 12pm
5	Examination Venue	MB/07A
6	Seat Number	3
7	Number of Answer Books handed in	

List the numbers of the questions you have attempted	Office Use
1	~~19~~ 20
2	18
3	19
Office Use	77%

$$\begin{array}{r} 575 \\ \hline 75 \end{array}$$

77%

GT 1/6/18

Question 1

a) TypeScript acts as a JavaScript preprocessor in the same way that SASS and LESS act as CSS preprocessors; it provides a user-friendly approach to writing object-oriented JavaScript.

Classes work in a similar way to other languages - the filename matches the class name, and contains the attributes (e.g "private _x: number"), constructor (e.g "constructor(x: number, y: number)"), properties (e.g "get X(): number") and methods (e.g "Draw(): void").

Interfaces provide a "contract" that obliges any class implementing it to provide the methods created in the interface. In this case, "Shape implements IDrawable" ensures that the Shape class specifies a Draw() method.

Abstract classes and methods provide a template that is intended to be overwritten (polymorphism) or extended by child classes. The presence of an abstract method means that the class must be made abstract - this is shown in the "abstract Draw(): void" and the "abstract class Shape". Abstract methods cannot be invoked by the abstract class, for reasons explained below.

Inheritance is shown in "class Rectangle extends Shape" - Shape is a generic class, from which multiple shapes can be drawn, each with their own properties. It therefore makes no sense to call the Shape class directly; when a specific shape needs to be drawn, the respective class can be called which inherits properties from the Shape class.

b) At present, the class only allows for a rectangle to be drawn. Using this to allow for a picture, it could be modified as follows:

```
class Picture extends Shape {  
    private _shapes[]: Rectangle[];  
  
    constructor(el: JQuery, position: Position) {  
        super (el, position);  
    }  
  
    AddRectangle(rectangle: Rectangle): Rectangle[] {  
        rectangle: Rectangle = new Rectangle;  
        this._shapes.push(rectangle);  
        return this._shapes[];  
    }  
}
```

3-5

```
Draw(): void {  
    this.El.html("Draw " + this._shapes[]);  
}  
}
```

X for loop required

c) i) XML

```
<employee>  
    <name>Fred Bloggs</name>  
    <department>Engineering</department>  
    <roles>  
        <role1>Designer</role1>  
        <role2>Programmer</role2>  
        <role3>Tester</role3>  
    </roles>  
    <email>fred.bloggs@company.com</email>  
    <telephone>01523847983</telephone>  
</employee>
```



ii) JSON

```
{  
  "employee": {  
    "name": "Fred Bloggs",  
    "department": "Engineering",  
    "roles": ["Designer", "Programmer", "Tester"],  
    "email": "fred.bloggs@company.com",  
    "telephone": "01523847983"  
  }  
}
```

8



Question 2

a) i) REST works by making use of the HTTP requests that have existed since inception, but only two of which are commonly used. It maps these requests to CRUD (Create, Read, Update, Delete) operations as follows:

- Create - POST
- Read - GET
- Update - PUT
- Delete - DELETE

REST sends data from service to application (and vice versa) in either XML (eXtensible Markup Language) or JSON (JavaScript Object Notation) format. Both of these are universal formats that can be managed by any language (e.g. PHP's json.encode / decode) to translate that data into a usable object.

The idea of this is to separate the server-side functionality (i.e. accessing a database) from the client-side (displaying the data). It takes some extra load off the server and lets the client's browser handle the interaction between user and website.

ii) AJAX stands for Asynchronous JavaScript And XML, although this is somewhat misleading, as it supports either XML or JSON. In fact JSON is the preferred format, as it is lighter, easier to read and write, and is more efficiently parsed by the client and service.

The key part of AJAX is "asynchronous" - it allows for data to be updated on a web page without the need to refresh the whole page. It can be used with JavaScript frameworks such as Sammy to create a purely client-side navigation structure; it can also be used to dynamically update text or other elements as a user types (this is often seen in search forms used by apps like WordPress).

b) By default, browsers do not allow for cross-origin access (hosted on different servers / domains to that which hosts the service) - all data must come from the same domain that provides the service. This is a security feature to prevent hackers from intercepting the signal as it's transmitted, and injecting malicious code that could, for instance, steal or corrupt data.

There are a number of ways round this, the most efficient being CORS (Cross Origin Resource Sharing), which is supported by all modern browsers. It involves setting permissions to be included in the request header, which would allow certain services (or all, by use of the * wildcard) to access the service from a different domain / server.

Question 3

a) The 4 Vs of big data are:

- Velocity: Refers to the speed and overall efficiency of data processing and general management.

- Voracity: Refers to the quality of the data being handled; i.e. is it reliable, correct, complete, corrupted...
- Variety: Refers to the many different formats that data can take - not just in terms of management systems (i.e. MongoDB, NoSQL, etc.), but also text, images, video, etc.
- Volume: By its very name, big data means the processing of vast quantities of data - currently this runs into TBs or even PBs of data.

b) The data being received comes from many different sources, and is fragmented. An example of this is shown below:

- Name: Fred, Age: 21, Occupation: Student
- Name: Fred, Likes: Gaming, Football, Gender: Male
- Name: Fred, Age: 21, Lives in: Swansea

All the incoming data is stored as-is, to be managed by the rest of the system.

The batch layer is where incoming data is stored; that is its only job. It is not concerned with the processing of data - that is the job of the service layer, which pulls the relevant information from the batch layer for running queries.

However, because of the amount of data being managed, running queries against it is a time-consuming process; therefore the results shown in the service layer - the batch view - are likely to be out of date by the time they are consulted.

To work around this, there is the speed layer - its job is to sort through the newest data received and present them in a real-time view that can also be accessed by queries. It provides a bridge between the newest incoming data and that stored in the batch layer.

The query itself takes the form of "MapReduce"; it works by querying the amount of times key words appear in the data, with the results appearing like this:

Fred - 1

Fred - 1

Fred - 1

These occurrences are mapped to the various fragments of data that are received, before being reduced to a word count (Fred - 3). By mapping the data in this way, a detailed picture can be built of the users whose data is being stored and what services they use.

c) One reason the data being stored is so large is because it never gets deleted. This is to allow for the likelihood of errors; if the voracity is compromised, there is still the older data to refer to. The constant

processing of MapReduce ensures that it is still possible to ensure consistency of data, with newer data being accessible in the speed layer when it arrives.

3

✓
19