INF4375 – Paradigmes des échanges Internet

Examen intra – 11 juin 2013 17:30 à 20:30

Nom, prénom :

Code permanent :

Répondez directement sur le questionnaire. Aucune documentation n'est permise.

**Question #1 – 10%**

Traduisez le document JSON suivant en XML. Votre document XML doit avoir la même modélisation que le document original.

```
{
  "capsule": "Légume",
  "ingredients": [
    {
      "nom": "Suma",
      "mg": 100
    },
    {
      "nom": "Sélénium",
      "mcg": 50
    },
    {
      "nom": "Vitamine A",
      "UI": 6000
    }
  ]
}
```

.....

.....

.....

.....

.....

.....

.....

.....

**Question #2 – 8%**

Indiquez si le document XML suivant est valide et bien formé. S'il ne l'est pas, expliquez pourquoi.

```
<?xml version="1.0" encoding="UTF-8"?>
<home><room name="kitchen"><oven /><table /><refrigerator><milk>
</refrigerator></room><room><name>Living room</name><carpet />
</room></home>
```

.....

.....

.....

.....

.....

.....

.....

.....

**Question #3 – 15%**

Écrivez le document HTML qui sera généré par le document Jade suivant. Indentez votre HTML pour en faciliter la lecture, même si Jade n'indente pas le code généré.

```
!!!
html
  head
    meta(charset="utf-8")
    title Une longueur d'avance
      | sur les autres participants
    link(rel="stylesheet", href="style.css")
    script
      function ilovejs() {
        return "J'adore le Javascript";
      }
  body
    #main-content
      p#message.text-success.result ilovejs();
      p.text-error Emplacement des erreurs
```

**Question #4 – 8%**

Qu'est-ce qu'une «route» selon Express.js?

.....

.....

.....

.....

.....

.....

.....

.....

**Question #5 – 8%**

Est-il possible de supprimer une propriété d'un objet Javascript (*Javascript – Gardez le meilleur, chapitre 3*)? Si non, expliquez pourquoi. Si oui, donnez un exemple où il serait utile de faire une telle opération.

.....

.....

.....

.....

.....

.....

.....

.....

**Question #6 – 8%**

Il existe une particularité assez problématique dans le fonctionnement de la propriété *length* de l'objet Array en Javascript (*Javascript – Gardez le meilleur, chapitre 6*). Cette particularité rend la propriété peu fiable. Décrivez cette particularité.

.....

.....

.....

.....

.....

.....

.....

.....

**Question #7 – 8%**

Node.js a été conçu de façon à gérer les entrées et sorties de façon purement asynchrone. Quels sont les avantages et inconvénients d'un tel design?

.....

.....

.....

.....

.....

.....

.....

.....

**Question #8 – 8%**

Considérant le document XML suivant :

```
<?xml version="1.0" encoding="utf-8"?>
<root xmlns:guo="http://www.tinaguo.com/"
      xmlns:rrs="http://reallyrightstuff.com/"
      xmlns="http://jberger.org/">
  <rrs:const xmlns:sst="http://www.sstgroupeconseil.com/">
    <guo:neee>Texte</guo:neee>
  </rrs:const>
  <sst:express xmlns:sst="http://www.cchst.ca/oshanswers/">
    <tri>
      <sst:laurier>Valeur</sst:laurier>
    </tri>
    <rrs:ph>map</rrs:ph>
  </sst:express>
  <js xmlns="http://www.w3.org/">Javascript</js>
</root>
```

Quel est la valeur du namespace (et non pas le préfixe) de chacun des éléments suivants :

- root :
- const :
- neee :
- express :
- tri :
- laurier :
- ph :
- js :

**Question #9 – 8%**

Quels sont les avantages d'offrir des services REST plutôt que des services SOAP?

.....

.....

.....

.....

.....

.....

.....

.....

**Question #10 – 8%**

Expliquez comment Ajax permet de diminuer la charge de travail d'un serveur web par rapport à une application web traditionnelle (sans Ajax).

.....

.....

.....

.....

.....

.....

.....

.....

**Question #11 – 11%**

Expliquez ce que fait le module Node.js suivant.

```
var http = require("http");

exports.mysteriousFeature = function (callback) {
  var options = {
    host: "www.uqam.ca",
    method: "GET"
  };
  var request = http.request(options, function (result) {
    if (result.statusCode !== 200) {
      callback("HTTP Error: " + result.statusCode);
    } else {
      var chunks = [];
      result.setEncoding("utf-8");
      result.on("data", function (chunk) {
        chunks.push(chunk);
      });
      result.on("end", function () {
        var completeData = chunks.join("");
        var list = completeData.match(/<div class="post">/g);
        callback(null, list.length);
      });
    }
  });
  request.on("error", function (e) {
    callback(e);
  });
  request.end();
}
```

.....

.....

.....

.....

.....

.....

.....

.....