Fachhochschule Bonn-Rhein-Sieg

Fachbereich Angewandte Informatik

Matrikelnummer: **Musterlösung**

Fachprüfung: Hyper- und Multimedia

Hypermedia- und Multimediasysteme (1,2)

29.09.2003

Name (bitte in Blockschrift)	
Matrikelnummer	
Unterschrift	

Hinweise:

- Überprüfen Sie Ihr Klausur-Exemplar bitte vor Beginn der Klausur auf Vollständigkeit.
- Bitte halten Sie Ihren Lichtbildausweis sowie den Studentenausweis zur Kontrolle bereit.
- Dauer der Klausur: 120 min.
- maximal erreichbare Punktzahl: 200
- Bitte füllen Sie das Deckblatt vollständig aus, beschriften jedes Blatt mit Ihrer Matrikelnummer und unterschreiben Sie dieses Klausur-Exemplar.
- Jedes Verlassen des Prüfungsraums muss ausdrücklich mit der Aufsicht vereinbart werden.
- Zugelassene Hilfsmittel sind ausschließlich Schreibutensilien, nicht-programmierbare Taschenrechner und das eigene(!) Gedächtnis.
- Bitte vermeiden Sie die Verwendung von roter Farbe.

- Die nach jeder Frage eingeklammerte Zahl ist die bei dieser Frage maximal erreichbare Punktzahl.
- Beachten Sie die in vielen Fragen enthaltenen Teilfragen!
- Falls der Platz für die Beantwortung einer Frage nicht ausreichen sollte, verwenden Sie bitte die Rückseite.
- Nutzen Sie im Falle von Unklarheiten hinsichtlich der Fragestellung die Möglichkeit zu Rückfragen!

Viel Erfolg!

Punkte Teil 1	
Punkte Teil 2	
Punkte gesamt	

1. Prüfer	..... Note
2. Prüfer	

Matrikelnummer: _____

– I. Prüfungsfragen zur Veranstaltung im SS 2002 (HMS-1) –

1. Nennen Sie typische Eigenschaften von Scriptsprachen für HTML. Unterscheiden Sie Client- und Server-seitige Sprachen (z.B. Vor-/Nachteile) und nennen Sie je 2 Beispiele. *(15)*

typische Eigenschaften:
- interpretative Ausführung *(++)*
- kaum bis keine Typisierung *(+)*
- keine komplexen Datenstrukturen *(+)*
- integrierte Betriebssystem-Kommandos *(+)*
- unterstützte String-Manipulationen *(+)*
- einbinden externer Programme *(+)*
- ...

Client-seitig:
- kein Verbrauch von Server-Ressourcen *(+)*
- geringere Bandbreite der Leitung erforderlich *(+)*
- bessere Datensicherheit *(+)*
- offline möglich *(+)*
- Bsp.: JavaScript, VBScript, ... *(++)*

Server-seitig:
- kein Verbrauch von Server-Ressourcen *(+)*
- Datenaustausch möglich, inkl. Datenbank-Anbindung *(+)*
- Verbergen von Quellcode einfacher *(+)*
- Plattform-unabhängig *(+)*
- Bsp: JScript, Perl, PHP, ASP, ... *(++)*

2. Wozu können HTML-Kommentarzeichen innerhalb eines JavaScript-Tags dienen? (siehe Beispiel) *(4)*

```
<script LANGUAGE="JavaScript">
<!-- function brmpf() { // code } // -->
</script>
```

Auskommentieren der Script-Inhalte für JavaScript-unfähige Browser *(++)*, da sonst Tag-Inhalt als Text *(++)* ausgegeben würde.

3. Welche (vier) verschiedenen Möglichkeiten bestehen zur Einbettung von JavaScript in HTML-Dokumente. Zeigen Sie Beispiele. *(12)*

- über eigene Datei: *(+)*
  <script LANGUAGE="JavaScript“ src=“JSdatei.js"
  type="text/javascript"></script> *(++)*
- als Markup im Script-Tag: *(+)*
  <script LANGUAGE="JavaScript">
  function brmpf() { // code }
  </script> *(++)*
- als URL im Anchor-Tag: *(+)*
  <a Href="javascript:alert('Hallo')"> Sag was *(++)*
- als Entity in regulärem HTML-Tag: *(+)*
  <hr Width=&{breite};> *(++)*

4. Wie werden in JavaScript mehrdimensionale Arrays erzeugt und initialisiert? *(6)*

1. Erzeugung eines eindimensionalen Arrays über Konstruktor:
   new Array() *(++)*
2. Funktionsdefinition: *function ...() {}* ; *(++)*
   enthält geschachtelte *for*-Schleifen mit jeweils neuen Konstruktor-Aufrufen
3. Initialisierung durch Funktionsaufruf *(++)*

5. Warum kann Perl zugleich als Interpreter- und als Compiler-Sprache bezeichnet werden? *(3)*

Hochsprachiges Script wird bei Aufruf durch Interpreter compiliert und erst danach ausgeführt *(+++)*

6. Welche Stationen durchläuft ein Java-Applet vom Quellcode bis zur Hardware-Ausführung? *(6)*

1. Compiler --> ByteCode *(++)*
2. ByteCode Loader *(+)*
3. ByteCode Verifyer *(+)*
4. ByteCode Interpreter (--> Laufzeitsystem) *(++)*
   od. jit-Compiler (--> Maschinencode) *(++)*
5. Ausführung

7. Warum können Sie beim Programmieren von Java Applets auf eine `main()` Methode verzichten? Über welche Anbindung wird die grafische Ausgabe gesteuert? *(5)*

- Applets nutzen die grafische Oberfläche des Browsers; deshalb müssen Ein- und Ausgabe-Funktionalitäten nicht im Programm selbst gesteuert sondern nur über die Browser-Schnittstelle angesprochen werden. *(+++)*
- Grafische Ausgabe erfolgt über ein Graphics Objekt (vgl. awt-Package), welches das Browser-Fenster anspricht. *(++)*

8. Beschreiben Sie zwei grundsätzlich verschiedene Möglichkeiten, Threads in Java zu programmieren. Gehen Sie dabei auch besonders auf die Unterschiede ein. *(10)*

Thread-Klasse: *(+)*
- eigene Klasse extends Thread*(++)*
- run() Methode wird durch start() aufgerufen *(++)*

Runnable-Interface: *(+)*
- eigene Klasse implements Runnable*(++)*
- start() Methode muss programmiert werden *(++)*

9. Erläutern Sie das Prinzip der Synchronisation beim „Multithreading“ in Java-Programmen. *(7)*

- Synchronisation: Vermeidung gleichzeitigen Zugriffs auf gemeinsame (veränderliche) Speicherbereiche/Variablen. *(+++)*
- Objekt-Monitore *(+)* beobachten den Zustand synchronisierter Methoden (als “`synchronized`“ deklariert) *(+)* und regeln den Zugriff.
- Über die Methoden `wait()` und `notify()` können synchronisierte Methoden direkt miteinander kommunizieren. *(++)*

10. Nennen Sie vier verschiedene Java Layout Manager und beschreiben Sie kurz deren Eigenschaften. *(12)*

- FlowLayout *(+)* – Anordnung der Elemente zuerst horizontal, dann vertikal *(++)*
- BorderLayout *(+)* – Anordnung der Elemente nach Ortsangabe [Center, North, West, East, South] *(++)*
- GridLayout *(+)* – Anordnung der Elemente nach Reihenfolge des Auftretens innerhalb eines vorgegebenen Gitters mit gleichmäßig großen Zellen *(++)*
- CardLayout *(+)* – Anordnung der Elemente hintereinander in z-Ebene mit Karteireitern *(++)*
- GridBagLayout *(+)* – Anordnung der Elemente innerhalb eines frei konfigurierbaren Gitters *(++)*
- BoxLayout *(+)* – Anordnung der Elemente nach Reihenfolge des Auftretens untereinander *(++)*

11. Beschreiben Sie das Prinzip und den Vorteil des double-buffering. Wie unterscheiden sich diesbezüglich die Java-Pakete AWT und Swing? *(12)*

- double buffer: Speicherbereich mit schnellem Zugriff für grafische Ausgabe wird für den gesamten Bildschirm (frame buffer) doppelt angelegt *(++)*. Während der Inhalt des einen (front buffer) angezeigt wird *(++)*, erfolgt das Rendern des aktuellen Bildinhalts (frame) im Hintergrund in den anderen *(++)*. Nach Abschluss des Renderns im back buffer werden beide ausgetauscht *(++)*. Der Vorteil besteht darin, dass der Bildaufbau selbst nicht sichtbar wird *(++)*.
- Swing verwendet für Animationen automatisch double buffering, während dieses in AWT eigens programmiert werden muss. *(++)*

12. Beschreiben Sie das Konzept der keyframe Animation („tweening“) in Macromedia Director oder Flash. Welche Eigenschaften lassen sich auf diese Weise animieren? *(8)*

- Einzelne Bilder werden als *Sprites* in bestimmtem Zustand bestimmten Zeitpunkten (*frames*) zugeordnet, Zu denen sie Schlüsselzustände annehmen (--> *keyframes*). *(++)*
- Für die zwischen den *Keyframes* liegenden Einstellungen werden die jeweiligen Zustände (für jeden *frame*) linear interpoliert und so eine kontinuierliche Veränderung erreicht. *(++)*
- Folgende Eigenschaften von Sprites können „getweent“ werden: Position, Orientierung, Größe, Form, Farbe, ... *(++++)*

Punkte Teil 1	/ 100

– II. Prüfungsfragen zur Veranstaltung im SS 2003 (HMS-2) –

1. Was sind „DataIslands“, wie und in welchem Kontext werden sie verwendet? Wie kann auf „DataIslands“ zugegriffen werden? *(10)*

Mit Hilfe von DataIslands können XML-Daten direkt innerhalb einer HTML-Seite enthalten sein. Der Vorteil ist dabei, dass die Daten dem Client zur ständigen Verfügung stehen, sobald die betreffende Seite im Browser geladen ist. Eine mögliche Anwendung ist zum Beispiel das Verwalten von Tabelleninhalten, die auf Anforderung hin umsortiert werden sollen ohne dass die Seite oder die Informationen in der Tabelle erneut vom Server geladen werden sollen.

Eingebunden werden können DataIslands u.a. wie folgt:

```
<XML ID="XMLID">
 <XMLDATA>
  <DATA>TEXT</DATA>
 </XMLDATA>
</XML>

oder

<XML SRC="http://localhost/xmlFile.xml"></XML>

oder

<SCRIPT LANGUAGE="XML" SRC="http://localhost/xmlFile.xml"></SCRIPT>

oder

<SCRIPT ID="XMLID" LANGUAGE="XML">
 <XMLDATA>
  <DATA>TEXT</DATA>
 </XMLDATA>
</SCRIPT>
```

Der Zugriff geschieht dann ganz normal mit JavaScript, z.B.:

```
function returnXMLData()
{
return document.all("XMLID").XMLDocument.nodeValue;
}

function returnXMLData()
{
return XMLID.documentElement.text;
}
```

2. In DHTML gibt es die Properties „innerHTML“, „outerHTML“, „innerText“ und „outerText“. Diese können lesend und schreibend verwendet werden. Beschreiben Sie die jeweiligen Unterschiede anhand des gedachten Beispieles „Das Fernsehprogramm des WDR“ und einer Zuweisung „Neuer Text“. *(10)*

- Lesen
  - innerHTML : Das Fernsehprogramm des WDR
  - outerHTML: Das Fernsehprogramm des WDR
  - innerText und outerText: Das Fernsehprogramm des WDR
- Schreiben
  - innerHTML: Neuer Text
  - outerHTML: Neuer Text
  - innerText: Neuer Text
  - outerText: Neuer Text

3. Welche Aufgaben erfüllt das Textrange-Objekt in DHTML? *(5)*

Das Textrangeobject dient umfangreicheren Operationen mit Text im HTML-Stream:

- Text in einem bestimmten Kontext (Element oder Position) zu identifizieren;
- Text im Dokument zu suchen, selektieren und ersetzen;
- Text in logische Einheiten zu zerteilen; Text und HTML zu manipulieren.

4. Wie lassen sich im MS Internet Explorer eigene Elemente definieren und anzeigen? Schreiben Sie ein Beispiel. *(10)*

Über die Definition eines Namesspace „xmlns“ im HTML-Tag und der Deklaration entsprechender Styles für die innerhalb des Namespace vorkommenden Tags

```
<HTML xmlns:FH="urn:FH">
  <HEAD>
        <STYLE>
        @media all{FH\:SEMESTER{COLOR:RED;}}
        </STYLE>
  </HEAD>
  <BODY>
        <FH:SEMESTER>6. Semester</FH:SEMESTER>
  </BODY>
</HTML>
```

5. Nennen Sie typische Bestandteile von DHTML und deren Funktion. *(10)*

Das Dokumentenobjektmodell (DOM) beschreibt den hierarchischen Aufbau der Elemente einer HTML-Seite, auf die (und deren Eigenschaften und Methoden) mit Hilfe von Scriptsprachen (dynamic content) wie JavaScript und Cascading Style Sheets (CSS) (dynamic styles) zur Laufzeit lesend und schreibend zugegriffen werden kann. Events schließlich dienen der Möglichkeit, auf Benutzeraktionen (button click, scrolling, ...) mit Hilfe des Scriptings reagieren zu können.

6. Auf welche (4) verschiedenen Arten können Cascading Stylesheets (CSS) in HTML Seiten verwendet werden? Lassen sich die unterschiedlichen Arbeitsweisen in Bezug auf ein HTML-Element kombinieren? *(10)*

Inline über das Attribut Style auf Ebene des einzelnen Elementes, Embedded im Style-Tag im Headbereich, External über das Link-Tag als Verweis auf eine separate CSS-Datei oder per Scripting. Die genannten Formen lassen sich mischen bzw. kombinieren.

Dabei gilt folgende Priorität in absteigender Reihenfolge bei sich widersprechenden Anweisungen: Scripting > Inline > Embedded > External.

7. Was sind die wesentlichen Unterschiede zwischen XML-Schema und DTD's? Wofür werden sie verwendet? *(5)*

Schemas werden selbst auch in XML geschrieben, Element- und Attributinhalte lassen sich näher typisieren. Das Contentmodell lässt sich gezielter, aber auch flexibler beschreiben.

8. Gegeben sei folgender Ausschnitt eines HTML-Dokumentes: „<U>Das ist Text mit einem
 Zeilenumbruch</u> > *und abschließender horizontaler Linie<hr>*". Ist das Beispiel auch ein XML-Dokument? Begründen Sie Ihre Entscheidung und schreiben Sie das Beispiel ggf. „richtig" um. *(10)*

Der Ausschnitt ist nicht wellformed. Für xHTML gilt u. a.: case-sensitive, schließende Tags, geordnetes Nesting, Attributwerte in Anführungszeichen, Script- und Styleblöcke in CData-Abschnitten.

Richtig wäre:
<u>Das ist Text mit einem
 Zeilenumbruch</u> und abschließender horizontaler Linie<hr/>

9. Wie lassen sich Datenbankabfragen mit ADO weiterverarbeiten? *(10)*

> Die Ergebnisse von SQL-Abfragen über ADO sind zunächst in einer temporären Tabelle enthalten („recordset“). Die Inhalte können in einer („for-while“) Schleife für die einzelnen Datensätze sowie einer Schleife über die Feldinhalte ausgelesen und weiterverarbeitet werden. Der Inhalt kann aber auch als XML-Stream („recordset.Save(xml, adPersistXML)“) oder als String („Variant = recordset.GetString(StringFormat, NumRows, ColumnDelimiter, RowDelimiter, NullExpr)“) ausgegeben werden.

10. Gegeben sei folgendes XML-Dokument. Beschreiben Sie mindestens 1 Befehl, um die Person zu ermitteln, die als Nachnamen „Müller“ hat. In welchem Kontext dann der Befehl verwendet werden? *(10)*

```
<?xml version="1.0"?>
<root>
        <person>
        <nachname>Noelle</nachname>
        < vorname>Guido</vorname>
        </person>

        <person>
        <nachname>Müller</nachname>
        < vorname>Karl</vorname>
        </person>
</root>
```

> //root/person[nachname =”Müller”]
>
> Im Dom-Modell mit
>
> xmldoc.selectSingleNode(befehl);
>
> oder
>
> xmldoc.selectNodes(befehl);
>
> oder in XSLT, z.B. <xsl:value-of select=”befehl” />

11. Was sind Namespaces, wozu werden sie benötigt, wie werden sie deklariert? *(10)*

Namespaces sollen ermöglichen, dass Dokumente aus verschiedenen Quellen gemeinsam verarbeitet werden können. Sie vermeiden Überschneidungen bei formal identischen Namen von Elementen oder Attributen, die aber inhaltlich unterschiedlich zu betrachten sind. Beipiel: "Titel" kann ein akademischer Titel oder ein Buchtitel oder ein Pfändungsanspruch sein.

Zur Unterscheidung werden vor die lokalen Bezeichner Präfixe gesetzt, die die lokale Bezeichnung identifizieren, sog. "Qualified Names".

```
<?xml version='1.0'?>
<form xmlns:FH="urn:fh">
    <FH:field>
        <content>Spezialfeld</content>
    </FH:field>
    <field>Normales Feld</field>
</form>
```

Punkte Teil 2	/ 100

Viel Erfolg!