Fahrzeugtechnik

fachhochschule hamburg
*FACHBEREICH FAHRZEUGTECHNIK*
Studiengang Flugzeugbau

Berliner Tor 5
D - 20099 Hamburg

in co-operation with:

The Queen’s University of Belfast
Belfast
BT7 1NN
Northern Ireland
United Kingdom

Diploma Thesis
- aircraft construction -

# Computer Based Learning in Aircraft Design – A Case Study with HTML and JavaScript

Author: Thomas Perthel

Release Date: 08.01.2000

Advisor at QUB: Dr. Richard Cooper

1st examiner: Prof. Dr.-Ing. Dieter Scholz, MSME
2nd examiner: Prof. Dr.-Ing. Willy J. G. Bräunling

# Abstract

This document will inform about the programming of a computer based learning course module with the topic preliminary sizing in aircraft design. The module covers the subjects landing, takeoff, second segment climb and missed approach climb requirements. The whole course is based on the Internet document type HyperText Markup Language, its additional programming language JavaScript and the design property Cascading Style Sheets. Portrayed are special features of dynamic HTML, most important features of the JavaScript program and possibilities of corporate design with Cascading Style Sheets. The connections between them and HTML will be described; pros and cons are presented. Especially the features to compose a document more interactively, user-specialized and dynamic are described in detail and recommendations for a learning course are given. The latest documentation of JavaScript published by the Netscape Communications Corporation and its counterpart JScript published by the Microsoft Corporation were used. Further, parts of the newer HTML 4.0 recommendations from the World Wide Web Consortium are described, which sets important standards in the growing business 'Internet'. The clear explanations of the JavaScript source code are presented with the programmed learning course, like the calculation of parameters in the sizing process of an aircraft. For example, one of the topics is the query of values given by the user via text fields, checkboxes and radio buttons and the control of its validity. Further, the storage - possibility with the cookie function is explained; usage and facts about its security are presented. The described multimedia course is attached to this document on a CD-ROM.

fachhochschule hamburg

*FACHBEREICH FAHRZEUGTECHNIK*

Studiengang Flugzeugbau

Queen's University of Belfast

School of Aeronautical Engineering

# Computer Based Learning in Aircraft Design – A Case Study with HTML and JavaScript

*Diplomarbeit* in compliance with § 21 of ”Ordnung der staatlichen Zwischen- und Diplomprüfung in den Studiengängen Fahrzeugbau und Flugzeugbau an der Fachhochschule Hamburg”

## Background

Computer Based Learning (CBL) Modules can be offered on the Internet, local networks or CD-ROM. The advantage: students can learn in a multimedia environment, independently of university resources. Various approaches and programming techniques exist for the module design. One possibility is to apply HTML and JavaScript programming.

## Task

The diploma thesis will research the possibilities of programming a CBL module with HTML and JavaScript. The work is based on a module featuring the first steps in preliminary sizing in aircraft design based on an approach presented by LOFTIN in NASA Reference Publication 1060. The module includes the calculation of aircraft design parameters from landing, takeoff, second segment climb and missed approach climb requirements. Subtasks are:

- Discussion of possibilities in multimedia course design with HTML / JavaScript
- Summary of key JavaScript features useful in aircraft design education based on available literature
- CBL module programming (results to be added to the report on CD-ROM)
- Presentation of selected features of the module on preliminary sizing in aircraft design
- Discussion of pros and cons of multimedia course design with HTML / JavaScript.

The results have to be documented in a report. The report has to be written in a form up to internationally excepted scientific standards. The application of the German DIN standards is one excepted method to achieve the required scientific format.

## Erklärung

Ich versichere, daß ich diese Diplomarbeit ohne fremde Hilfe selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

..........................................................................................

Datum Unterschrift

## Preface

This work originated in Belfast at the Queen’s University in advance of a formal bilateral agreement under SOKRATES-ERASMUS being set up between the Queens University of Belfast and the University of applied science Hamburg.

I would like to thank the foundation ‘Flughafen Frankfurt Main Stiftung’ for their grant, which was uncomplicated and quickly.

I would also like to thank Dr. Richard Cooper, senior lecturer at QUB, for his support at the preparations and technical circumstances.

Belfast, December 1999 Thomas Perthel

# Contents

# List of Symbols

| | |
|---|---|
| $A$ | Aspect ratio |
| $B_s$ | Breguet range factor |
| $B_t$ | Breguet endurance factor |
| $b$ | Wing span |
| $c_D$ | Drag coefficient |
| $c_L$ | Lift coefficient |
| $D$ | Drag |
| $d$ | Diameter |
| $e$ | Oswald's airplane efficiency factor |
| $g$ | Acceleration of fall |
| $h$ | Height |
| $k$ | Statistic factor, constant |
| $l$ | Length |
| $L$ | Lift |
| $L/D$ | Lift to drag ratio |
| $M$ | Mach number |
| $m$ | Mass |
| $m/S_W$ | Wing loading |
| $N$, $n$ | Number |
| $R$ | Range |
| $S$ | Surface |
| $s$ | Distance |
| $T$ | Thrust |
| $v$ | Velocity, speed |
| $v_S$ | Stall speed |

## Greek Symbols

| | |
|---|---|
| *a* | Angle of attack |
| *g* | *a)* Runway slope, *b)* Climb gradient |
| *m* | Roll friction |
| *r* | Density |

## Indexes

| | |
|---|---|
| *APP* | Approach |
| *CLB* | Climb |
| *CR* | Cruise |
| *D* | Drag |
| *F* | Fuel |
| *L* | *a)* Lift, *b)* Landing |
| *LFL* | Landing field length |
| *LOF* | Lift-off |
| *M*, *max* | Maximum |
| *MA* | Missed approach |
| *PAX* | Passengers |
| *PL* | Payload |
| *TO* | Takeoff |
| *TOFL* | Takeoff field length |
| *TOG* | Takeoff ground roll |
| *t* | Total |
| *W* | *a)* Wing, *b)* Wind |

# List of Abbreviations

| | |
|---|---|
| AEO | All Engines Operating |
| ASDA | Accelerate Stop Distance |
| CSS | Cascading Style Sheets |
| FAA | Federal Aviation Administration |
| FAR | Federal Aviation Regulations |
| HTML | HyperText Markup Language |
| HTTP | HyperText Transfer Protocol |
| JAA | Joint Aviation Authorities |
| JAR | Joint Aviation Requirements |
| LDA | Landing Distance Available |
| LFL | Landing Field Length |
| MLW | Maximum Landing Weight |
| MTOW | Maximum Takeoff Weight |
| OEI | One Engine Inoperative |
| QUB | Queen's University of Belfast |
| RFC | Request for Comments |
| TODA | Takeoff Distance Available |
| TOFL | Takeoff Field Length |
| URL | Uniform Resource Locator |
| V1 | Decision speed |
| V2 | Takeoff speed |
| VEF | Engine Failure speed |
| VS | Stall speed |

# Register of Terms and Definitions

**Browser**
Browser is a colloquial name for software programs, which present HTML documents.

**Cascading Style Sheets**
"Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g. fonts, colors, spacing) to Web documents." (**W3C 1997**)

**Cookie**
"A cookie is a small piece of information stored on the client machine in the cookies.txt file." (**Netscape 1999**)

**ECMA**
"ECMA is an international, Europe-based industry association founded in 1961 and dedicated to the standardization of information and communication systems." (**ECMA 1998**)

**Eventhandler**
Eventhandler "are used with client-side objects in JavaScript to evoke particular actions." ... "The name of an event handler is the name of the event, preceded by 'on.' " (**Netscape 1999**)

**HyperText Markup Language**
"To publish information for global distribution, one needs a universally understood language, a kind of publishing mother tongue that all computers may potentially understand. The publishing language used by the World Wide Web is HTML (from HyperText Markup Language)." (**W3C 1997**)

**HyperText Transfer Protocol**
"The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands)." (**RFC 1945**)

**JavaScript**
"JavaScript is Netscape's cross-platform, object-based scripting language for client and server applications." (**Netscape 1999**)

**JScript**

"JScript is the Microsoft implementation of the ECMA 262 language specification. It is a full implementation, plus some enhancements that take advantage of capabilities of Microsoft Internet Explorer." (**Microsoft 1999**)

**Request for Comments**

Request for Comments intents to "... make available some information for interested people, or to begin or continue a discussion of an interesting idea, or to specify a protocol." (**RFC 0825**)

**Stall Speed**

Stall speed is "... the minimum steady flight speed ... at which the airplane is controllable...". (**FAR 25**)

**Tag**

Tags mark the parts of an HTML - document e.g. paragraphs, tables or quotations. The HTML language elements are standardized by W3C Recommendations.

**Uniform Resource Locator**

"Uniform Resource Locators are sequences of characters, i.e., letters, digits, and special characters ..." and "... used to ‘locate’ resources, by providing an abstract identification of the resource location." (**RFC 1738**)

# 1 Introduction

Today's new technologies create ways to get information and knowledge quicker, easier and more extensive. The world is going to be connected and communication around the world is no longer a traffic problem. These new connections, the 'Internet', offer a publishing possibility, which is based on an easy understanding language called HyperText Markup Language. HTML with its possibility to 'markup' text elements like headlines, paragraphs and quotations is a new type of presenting manifold information. In the course of development of the Internet, HTML got a programming language to extend its capabilities. This tool, introduced under the name 'JavaScript' by the Netscape Communications Corporation in 1990, allows simple programming to support and improve the presentation of the HTML document in a browser.

This diploma thesis will find out the possibilities of a computer based learning course written in HTML and JavaScript. The theme of the learning module will be a rapid method for estimating typical parameters of an aircraft in the design process. Based on a design process introduced by Laurence K. Loftin in NASA Reference Publication 1060 (**Loftin 1980**), the course contains the demands for an aircraft resulting from a consideration of:

- landing field requirements
- takeoff field requirements
- second segment climb requirements
- missed approach requirements.

If these basis requirements determined, the intermediate results come together in a matching chart, where main parameters of the designed aircraft can be gained. At the end, a more complex calculation yields further important aircraft parameters.

# 2 Possibilities in Multimedia Course Design

## 2.1 HTML

HTML- files are simple text files, which can be produced, read and updated without specific software. A text editor or HTML editor respectively is available under every operating system and allows a course creation at all platforms. HTML describes the composition of a text; it marks headings, paragraphs, quotations, lists, etc. with the help of marking elements (tags). Thus, it is possible to give a learning course a basic shape with recognizable elements. Links can connect and support the learning course with further knowledge data banks, backgrounds or details from all over the world. With this important feature the learning material has a wide-spread source and provides the user different views or methods.

Publications written in the language of the World Wide Web can be accessed via an Internet connection from everywhere. For example, students can learn independent of a specific class-room, library or computer system.

## 2.2 HTML and JavaScript

HTML with its scripting language JavaScript offer many different operations while presenting documents, independent if they are accessed local or global. The most important feature is the interactivity, which means that user can influence on the content of the document, its behavior and the design.

For use in the engineering sector, it is possible to calculate results under fixed formulas with changeable input values repeatedly. User can not only read text, but also use a formula and increase by it the learning effect. If the user types in values, a script can read these values on every possible action, e.g. the change of a given value or the click on a button. Is the value controlled by the script, it can be applied to a formula or, if it is out of range and not valid, the user can be informed via a dialog box. If the calculation is done, the results can be appearing in a further text field or as plain text in the document. With the calculation, there are further actions possible. The value can release the display of an image or explaining text. In the module enclosed to this document, the calculated results will also be shown in a chart. By it, the understanding of the effects of input parameter changing is much easier understandable than only a sight on numerals. HTML offers also a feature to store values, words or names for a later go on work. If basis calculations are done, the results can be saved and later, if knowledge that is more complex is imparted, further calculations together with the stored results can be calculated.

A possible feature is also the ability to show and change data of many different file types. Images or animations of different file sizes can replace each other e.g., at a click on a button depending on user resources or needs. If the computer is connected to the Internet, the location of the graphics, text files, data banks etc. must not be the same as the location of the actual document.

## 2.3 Cascading Style Sheets

The design of a learning course is the one of the main parts to its success. Uniform appearance of all parts of the course makes the understanding quicker and the managing much more easily. For instance, quotations can fitted in other text-color and causes more attention from the user in this way. Corporate design also provides recognition every time the learning module is used later.

To obtain HTML with the ability to design it's pages detailed and type-specific the additional feature Cascading Style Sheets was built. Three different CSS statement positions for every circumstance are possible. Cascading Style Sheets permit to define design standards for all documents and their elements in a separate file, to which all components can access via a link-instruction in the head of file and take over the defined settings. Modifications which made in this CSS - file have consequences in all linked documents and therefore allows an easy course design by simply editing one single file. Design statements can also made in the head of the file. These instructions overwrite the instructions from the separate file, so that exceptions are possible, e.g. for an introduction or an epilogue. At last, CSS statements can made direct in the text; they will overwrite all other statements and can be used for special text design as the quotation notations in bold, small caps.

# 3 Summary of HTML Features

From the varied components of the HTML language, two important parts, forms and layers, will be presented in detail. The form feature is the possibility to define areas, which content can be read and written by a JavaScript or send to a server computer connected via the Internet. The second feature 'layer' is an HTML tag to unite different parts of the document, e.g. text or images, which can be called and controlled separately.

## 3.1 Forms and its Elements

Forms consist of a beginning tag and an end tag. Every statement or other element between these two tags is part of the form. Several behavior statements, called attributes, can be made together with the start tag:

- name: This attribute gives the name, with which the form can be addressed.
- action: Action lays down the location at the Internet, where all data will be send.
- method: Fixes the way, data will be sending.

A typical form has the following structure:

```
<form name="form1" action="" method="post">
  <input type="radio" name="engine" value="2" checked>
  <input type="radio" name="engine" value="3" >
  <input type="radio" name="engine" value="4" >
  <input type="text" name="sfc" size="4" maxlength="4" value="16.2" >
  <input type="checkbox" name="ff2" checked>
  <input type="reset" value="reset form">
  <input type="button" value="calculate" onClick="calculate()">
  <input type="text" name="result" size="4" maxlength="4">
</form>
```

### 3.1.1 Buttons

```
<input type="button" value="calculate" onClick="calculate()">
```

This tag displays a push button in the document. With the Eventhandler 'onClick', a client-side script can be triggered.

- type: This attribute describes the input type as button.
- value: Values contains the name  that will be displayed on the button.

### 3.1.2 Text fields

```
<input type="text" name="sfc" size="4" maxlength="4" value="16.2" >
```

Text fields allow a handing over of numbers or strings between the user and the script. The attributes mean:

- type: This attribute describes the input type as text field.
- name: It contains the name under which the text field can be called through the script.
- size: This attribute describes the size or length respectively, the field will have been have.
- maxlength: This attribute describes the number of digits, which will be displayed.
- value: It contains the start value , which is displayed first time.

### 3.1.3 Checkboxes

```
<input type="checkbox" name="ff2" checked>
```

Checkboxes permit the user to make a yes/no decision. The attributes mean:

- type: This attribute describes the input type as checkbox.
- name: It contains the name under which the text field can be called through the script.
- checked: Lays down, if the checkbox is activated at the first time.

### 3.1.4 Radio buttons

```
<input type="radio" name="engine" value="2" checked>
<input type="radio" name="engine" value="3">
<input type="radio" name="engine" value="4">
```

Radio buttons allow the user to select between different possibilities. The attributes mean:

- type: This attribute describes the input type as radio button.
- name: It contains the name under which the radio can be called through the script. All radio buttons with the same name allow only one active option.
- value: It contains the value or string, which this radio button represents.
- checked: Fixes, if the checkbox is activated at the first time.

## 3.2 Layers

In HTML, text blocks or other elements can be grouped together in one tag. Netscape Navigator provides for the feature the 'layer' and 'ilayer' tags. The contents in this tag can be called via the layer's name, which was defined in the 'id' attribute. Several further properties can be assigned to these layers, such as the visibility. It offers the possibility to show or hide the layer in the actual document. The following example is a layer with the name "Layer1" and after the load of the document in the browser window not visible, although the layer takes his paragraph in the document:

```
<layer id="Layer1" visibility="hide">
  <img src="landline.gif" height="166" width="19">
</layer>
```

The layer's content is an image of the graph for the landing field requirement. A script that runs after the load of the document – when the layer has an instance – can change the properties. In this case, the visibility - property is changed to "show" when the script determines a correct saved landing field parameter in the cookie:

```
if(k1==1)
   {
    www=eval(.39*w-22);
    document.layers.Layer1.moveTo(Math.round(www),289);
    document.layers.Layer1.visibility="show";
   };
```

The second layer tag 'ilayer' has the difference, that the contents forms no paragraph when the layer is hidden. The using is similar to the 'layer' tag:

```
<ilayer id="Layer1" visibility="hide">
  <img src="landline.gif" height="166" width="19">
</ilayer>
```

Microsoft's counterpart for the Internet Explorer is the 'div' tag. The name of the layer is also determined in the 'id' attribute; the visibility can be defined in the 'style' attribute according to the following example:

```
<div id="Layer1" style="position:relative;display:none;top:-56px;">
  <img src="landline.gif" height="166" width="19">
</div>
```

The 'style' instructions have the meaning:

- position: defines the position of the layer in the document; with "absolute" the position statements refer to the upper left position of the whole document, "relative" refers the position statements to the predecessor element,
- display: defines, if the layer is shown or not,
- top: distance statement from the predecessor element to the layer's upper side,
- left: distance statement from the predecessor element to the layer's left side.

Distance statements are possible in different units; for instance, the relative unit "px" means pixel. If the 'div' layer is not shown in the document, it doesn't lay claim to an own paragraph. If the layer has to be shown, the 'style' attributes property 'display' can be changed as follows:

```
if(k1==1)
   {
    www=eval(.39*w-22);
    document.all.item("Layer1").style.posLeft=Math.round(www);
    document.all.item("Layer1").style.display="inline";
   };
```

To 'posLeft' will be assigned the result from the positioning calculation, rounded to a whole number. The 'display' property will be assigned the string "inline", it means that the layer appears in continuous order in the document.

# 4 Summary of JavaScript Features

## 4.1 Eventhandler

### 4.1.1 onClick

The definition of **Netscape 1999** is:

> *"Executes JavaScript code when a click event occurs; that is, when an object on a form is clicked. (A click event is a combination of the MouseDown and MouseUp events)."*

The Eventhandler 'onClick' is used in the course design to control the calculations and drawings of the graphs at all main sites. Its syntax is:

onClick="instruction".

The instruction is in the course "preliminary sizing" a function call. The 'onClick' is connected to the HTML element 'button', when the button is clicked by the user. It is used according to the following example:

```
<input type="button" value="calculate" onClick="calcu()">
```

The function "calcu( )", stored in the head of the HTML file, will be executed at the button click.

### 4.1.2 onMouseOut

The definition of **Netscape 1999** is:

> *"Executes JavaScript code when a MouseOut event occurs; that is, each time the mouse pointer leaves an area (client-side image map) or link from inside that area or link."*

The eventhandler 'onMouseOut' controls in the course "preliminary sizing" the text in the status line of the browser window. Its syntax is:

onMouseOut="instruction".

Because of the using of instructions, which are working properly only in individual browser types, all main sites were written twice. To navigate between these sites, the eventhandler is used at links to these pages. OnMouseOut changes the displayed message of the status line according to the example below as follows:

```
<a href="javascript:scookie()" ... onMouseout="status='';">
```

To the property 'status' of the object 'window' is a sting assigned, which will be displayed in the status line. With an empty string, the message shown before will be erased.

### 4.1.3 onMouseOver

The definition of **Netscape 1999** is:

> *"Executes JavaScript code when a MouseOver event occurs; that is, once each time the mouse pointer moves over an object or area from outside that object or area."*

To hide the script instructions from being displayed in the status line, an explanation of the link instead will be displayed, if the user moves the cursor over the link in the document. OnMouseOver changes the displayed message of the status line according to the example below as follows:

```
<a href="javascript:scookie()" ... onMouseover="status='set cookie';return
true;">
```

To the property 'status' of the object 'window' is a sting assigned, which will be displayed in the status line. The status line shows the message "set cookie". The display of the script instruction will be avoided with the return statement.

## 4.2 Statements

- break:
  This statement interrupts the execution of the "if ... else' and 'switch' alternatives and the 'while' loop. It is used as in the following example:

```
switch (z)
  {
   case "2" :
   w=eval(2*(1/ld+0.024));
   break;
   ...
```

- comment:
  It gives the possibility to insert comment in the script. Comments are ignored by the interpreter. A single line can be declared as comment with two slashes in front, several lines inside "/* */":

```
//get values from the form:
   v=document.form1.clmaxto.value;

/*example tables*/
```

- for:
  Creates a loop that consists of three optional expressions, enclosed in parentheses and separated by semicolons, followed by a block of statements executed in the loop. It is used as follows (**Netscape 1999**):

```
for(i=0;i<4;i++)
  if(document.form1.range[i].checked==true)
    z=document.form1.range[i].value;
```

- function:
  Declares a function with the specified parameters. Acceptable parameters include strings, numbers, and objects (**Netscape 1999**).

```
function calcu()
  {if(as==1){...
```

The function will be executed by calling it with the expression 'javascript:' or an eventhandler as follows:

```
<input type="button" value="calculate" onClick="calcu()">

<a href="javascript:scookie()" ... >
```

- if ... else
  Executes a set of statements if a specified condition is true. If the condition is false, another set of statements can be executed. (**Netscape 1999**) Using:

```
if(ww!=w)
   {
    alert("Cookies are deactivated on this computer!\n\nThe result could
not be saved.");
    return;
   }
else
   {
    alert("Wing loading = "+ww+" kg/m² saved.");
   };
```

- return:
  Exits from the current function and returns a value from that function. If omitted, the function does not return a value. (**Microsoft 1999)** This statement will be used for the value input control as follows:

```
if(range<1||range>100000)
   {
    document.form1.range.value="";
    alert("The range is invalid.\n\nIt should be at least 1.");
    return;
   };
```

- switch:
  Enables the execution of one or more statements when a specified expression's value matches a label. Continue execution until a break statement is encountered, or the switch statement ends. This means that multiple label blocks are executed if a break statement is not used. (**Microsoft 1999)** It will be used with takeoff thrust to weight calculation in the second segment climb and missed approach requirements and switches the engine number:

```
switch (z)
  {
   case "2" :
     w=eval(2*(1/ld+0.024));
     break;
   case "3" :
     w=eval((3/2)*(1/ld+0.027));
     break;
   case "4" :
     w=eval((4/3)*(1/ld+0.03));
     break;
  };
```

- var:
  Declares a variable, optionally initializing it to a value. (**Netscape 1999**) Several variables can be declared with one statement, separated by commas:

```
var as=1,k,w=-1,z;
```

## 4.3 Objects

### 4.3.1 document

- cookie.indexOf:
  Returns the character position where the first occurrence a substring occurs within the saved cookie string. In the following example the position of "g" is stored at the variable "end":

```
end=document.cookie.indexOf("g");
```

- cookie.substring:
  Returns the substring at the specified location within the saved cookie string. The substring in the cookie between the location of "start" and "end" will be returned as follows:

```
ww=document.cookie.substring(start,end);
```

The string, which is saved in the cookie, can be only in the ASCII format; therefore cookies cannot contain executable programs like viruses. Also no space is allowed in the string.

- images.src:
  It represents the string specifying the URL of an image to be displayed in a document. It is used for the control of the animations as in the following example:

```
document.images.lflani.src="lafi.gif";
```

- location:
  It represents the string specifying the URL of the current document displayed in the browser window. It is used for the navigation between and within the HTML documents as the following example:

```
document.location="#bottom";
```

- layers.moveTo:
  Moves the top-left corner of the layer to the specified screen coordinates in the parentheses. It is used in the Netscape documents for the control of graph display:

```
document.layers.Layer1.moveTo(Math.round(y),-55);
```

- replace( ):
  The replace method loads the specified URL over the current history entry. After calling

the replace method, the user cannot navigate to the previous URL by using browser's Back button. (**Netscape 1999**) It is used in the course to navigate between the documents:

```
top.location.replace("lfl_nn.htm");
```

- value.all.item:
  This property represents the value of the document item specified in the parentheses. It can be used only in the Internet Explorer to address the layers:

```
document.all.item("Layer1").style.posLeft=Math.round(y);
```

- visibility:
  This property decides whether the layer is displayed or not. The string "show" means show the layer, "hide" means hide the layer. It is used in the Netscape documents as follows:

```
document.layers.Layer1.visibility="show";
```

### 4.3.2 Math

- PI:
  This property represents the mathematical constant 'ð' with its value approximately 3.14159. It is used for the lift to drag ratio calculation:

```
ld=eval(v/(v1+(v*v/Math.PI/w/0.7)));
```

- round:
  Returns the value of a number rounded to the nearest integer. (**Netscape 1999**) The coordinates for the graph display are rounded according to the example:

```
document.layers.Layer1.moveTo(Math.round(y),Math.round(z));
```

- sqrt( ):
  This method returns the square root of the number enclosed in the parentheses. It is used as follows:

```
v=eval(Math.sqrt(v)*1.7);
```

### 4.3.3 navigator

- appName:
  This property represents the name of the browser. It can be read only. In Netscape Navigator, the property is "Netscape", in the Internet Explorer "Microsoft Internet Explorer". It is used in the course to navigate between the documents:

```
if(navigator.appName=="Netscape")
   {...
```

## 4.4 Methods

### 4.4.1 alert( )

The 'alert' method displays a message box with an "OK" button in front of the actual browser window. That means that the user has to take notice of the string, enclosed in the parentheses of the method, before the execution of the script continues. The string is not HTML and can be formatted by using escape sequences from table 1.

**Table 1** Escape Sequences (**Microsoft 1999**)

| Escape Sequence | Character |
|---|---|
| \b | Backspace |
| \f | Form feed |
| \n | Line feed (newline) |
| \r | Carriage return |
| \t | Horizontal tab (Ctrl-I) |
| \' | Single quotation mark |
| \" | Double quotation mark |
| \\ | Backslash |

The following example shows the using of the 'alert' method in a script:

```
alert("Wing loading = "+ww+" kg/m² saved.");
```

### 4.4.2 confirm( )

The 'confirm' method displays a message box with an "OK" and an "Cancel" button in front of the actual browser window. That means that the user has to make a yes/no decision, then the execution of the script continues. The string in the parentheses, which will be displayed in the message box, can be formatted with escape sequences from table 1. The 'confirm' method is used as follows:

```
check=confirm("Save this?\n\nwing loading = "+w+" kg/m²");
```

Several strings can be connected with the symbol "+". In the example above, the answer from the user (true or false) will be saved at the variable 'check' and used later.

## 4.5 Functions

### 4.5.1 eval( )

The function 'eval' executes mathematical expressions and returns its result. If the string contains characters, which cannot be interpreted, it returns an error message. Mathematical expressions have to build with the operators from table 2.

**Table 2** Arithmetic Operators (**Netscape 1999**)

| Operator | Description |
|---|---|
| + | (Addition) Adds 2 numbers |
| ++ | (Increment) Adds one to a variable representing a number (returning either the new or old value of the variable) |
| - | (Unary negation, subtraction) As a unary operator, negates the value of its argument. As a binary operator, subtracts 2 numbers |
| -- | (Decrement) Subtracts one from a variable representing a number (returning either the new or old value of the variable) |
| * | (Multiplication) Multiplies 2 numbers |
| / | (Division) Divides 2 numbers |
| % | (Modulus) Computes the integer remainder of dividing 2 numbers |

According to the following example, the result from the expression in the parentheses is assigned to the variable "y":

```
y=eval(.39*w-537.91);
```

'eval' is a top-level function and is not associated with any object. (**Netscape 1999**)

### 4.5.2 isNaN( )

Returns a Boolean value that indicates whether a value is the reserved value NaN (not a number).(**Microsoft 1999**) This function is used with the cookie saving as follows:

```
if(isNaN(ww)==true)...
```

'isNaN' is a top-level function and is not associated with any object. (**Netscape 1999**)

# 5 Summary of Cascading Style Sheets Features

## 5.1 Definition Places

Basis definitions for a corporate design can be made in a separate CSS - file. This file is a text file, but with the ending ".css". Each HTML - document designed with the statements of this file has to have the following link declaration (**RFC 2068)** in its file head:

```
<link rel=stylesheet type="text/css" href="pre_siz.css">
```

The attributes have the meaning:

- rel: defines the relationship to this document,
- type: defines the MIME - file type of the link,
- href: path definition of the linked file.

CSS - statements in the head of an HTML - file are framed with the 'style' tag as follows:

```
<style type="text/css">
<!--
h1 { text-align:center; }
p,td,body { font-family:Helvetica,Arial; }
div { text-align:center;font-size:16pt; }
span { font-variant:small-caps; }
//-->
</style>
```

The tag '`<!--     //-->`' separates the Style Sheet definitions from the HTML definitions in the document, hides the data from non-conforming user agents and avoids error messages in this browsers. CSS - definitions in the text are made with the attribute 'style' in the beginning tag:

```
...R<span style="font-variant:small-caps;">oskam</span>...
```

If there is no tag available, parts of the text can be grasped with the 'span' tag, which has no special meaning than a style container (**W3C 1997**).

## 5.2 Statements

Style definitions are made with the specification of the tag to which they are assigned, separated through commas; and all statements in brackets of the type "{ }", separated through semicolons:

```
h1 { text-align:center; }
p,td,body { font-family:Helvetica,Arial; }
div { text-align:center;font-size:16pt; }
span { font-variant:small-caps; }
```

Classes can be defined for a tag that will be used in different ways. For instance, the paragraph tag 'p' gets a class named "reg" for quoting aviation regulations:

```
p.reg  {  width:90%;
          font-size:10pt;
          border-style:solid;
          border-width:2px;
          border-color:#000099;
          background-color:#FFFFCC; }
```

If the quotation begins in the HTML - file, the beginning tag is assigned the attribute 'class' as follows:

```
<p class="reg">
  JAR 25.125  Landing ...
</p>
```

With Cascading Style Sheets, many definitions are possible. The following extract describes only a small amount of available statements specified in **W3C 1997**.

### 5.2.1 Font formatting

- `font-family:Helvetica,Arial;`
  It determines the font of the specified text, if the font is not available, the next font in the list will be used.

- `font-size:16pt;`
  This statement defines the character height, "pt" means points (= 1/72 inch).

- `font-variant:small-caps;`
  It defines the specified text in the format small capital letters.

### 5.2.2 Frames

- `border-style:solid;`
- `border-width:2px;`
- `border-color:#000099;`

These statements define the design of borderlines around the accessory tag.

### 5.2.3 Margins

- `margin-bottom:3pt;`
- `margin-top:3pt;`

These statements define the distance of the accessory element to the document margin.

### 5.2.4 Positioning

- `text-align:center;`
- `vertical-align:middle`

These statements determine the text position in the document. The standard for 'text-align' is "left", for 'vertical-align' is it "top".

- `position:relative;`

It defines the position relation between the specific element and its environment, relative means that the element appears in continuous order in the document.

- `top:-56px;`
- `left:-415px`

These statements define the distance between the element and the document margins.

### 5.2.5 Display control

- `display:none;`

With the statement the visibility of the element in the browser window can be controlled, "none" means no visibility and no spacing, "inline" means the display in the text and "block" the display in an own paragraph.

# 6 Software Differences

Internet browsers are subjects to permanent changes and innovations. Especially the 'layer' feature, introduced only in Netscape Navigator 4.0, cause difficulties in programming HTML documents independent of the browser type. In the course "preliminary sizing in aircraft design", several of this features including the 'layer' feature were used. Therefore, it was necessary to create two different versions of all main course sites. Differences between these sites are listed in table 3. To avoid inadvertent loading of a Netscape HTML document into the Internet Explorer and reverse JavaScript checks the browser type and divert the browser to the appropriate document.

**Table 3** Programming Differences

| Element | Netscape Navigator 4.6 | Microsoft Internet Explorer 5.0 |
|---|---|---|
| layer | • Layers are created with the <layer> and <ilayer> tag • introduced by Netscape, not part of the HTML 4.0 specification by **W3C 1997** | • Layers are created with the <div> tag • manipulated with CSS |
| table | • formatting either in % or in pixel in a table | • formatting in % and pixel inside one table possible |
| anchor | • links and targets will be underlined | • only links will be underlined |
| input | • displayed only in forms • several buttons appear one below the other | • displayed everywhere in the document one behind the other |
| table borders | • not adaptable | • adaptable |
| CSS | • several 'text-align' statements will be misinterpreted • character height not adaptable for the <table>, <th> and <td> tags, bypassing with involved tags | • 'text-align' statements correct interpreted • character height adaptable for the <table>, <th> and <td> tags |
| type convert | • addition priority as character type, avoidance with multiplication in front | • mathematical addition is priority |
| Cookie | • all cookies saved in one file | • cookies saved separately |

# 7 CBL Module Programming

Every requirement is divided into its logical parts:

- Background:
  refers to the aviation regulations and gives explanations of the constraints that come from the regulations
- Equations
  explains the origin of all needed values, their connections and refers to statistics to get an impression of the meaning and the range
- Calculation
  allows a final calculation, which support the understanding of the parameters and the effect of its changes repeatedly. It follows a chart, which shows the requirement graphical for a later determination of the aircraft parameters.
- Results compared
  calculated results can be compared and checked against parameters from existing aircraft

Several links in the documents give the way to further and more detailed explanation, to back-grounds or data in the Internet.

## 7.1 Introduction

The starting page "start.htm" leads the preliminary sizing course. It explains the intention of the course preliminary sizing in aircraft design and gives an overview about the presented parts. Because of the possibility to view the requirement results graphical in a chart, two versions of each major page were created. From the starting page the main pages of the course depending on the used browser can be reached with the help of the following JavaScript:

```
function decide(x)
   {
    if(navigator.appName=="Netscape")
     {
      top.location=x+"_nn.htm";
     }
    else
     {
      top.location=x+"_ie.htm";
     };
   };
```

The document's property 'location' will be assigned the correct page name, when in the 'if' statement the navigator's property 'appName' is compared with the string "Netscape". The function 'decide( )' is called by the special expression 'javascript:' in the 'href' attribute of the anchor tag.

```
<a href="javascript:decide('lfl')"
         onMouseover="status='landing field length';return true;"
         onMouseout="status='';"
         target="_top">landing field length</a>
```

The 'onMouseover' and 'onMouseout' Eventhandler hide the JavaScript instruction from being displayed in the status line at the bottom of the browser window.

## 7.2 Landing Requirements

### 7.2.1 Page Layout

The quotations from the aviation regulations are displayed separately with light background and a blue frame line with the help of the following CSS instructions, where the class 'reg' is defined for the paragraph tag:

```
p.reg  {  width:90%;
          font-size:10pt;
          border-style:solid;
          border-width:2px;
          border-color:#000099;
          background-color:#FFFFCC; }
```

In the requirement landing field length evolves from four different basis equations a ground formula, which determines the wing loading:

$$\frac{m_{MTO}}{S_W} = \frac{k_L \cdot \frac{r}{r_0} \cdot c_{L,\max,L} \cdot s_{LFL}}{\frac{m_L}{m_{TO}}} \qquad (1.0)$$

The equation (1.0) is created as an imagemap, which allows a click on every part of the formula to get different information or explanations at an other page. The also displayed basis equations are links, which led to backgrounds (e.g. **Dubs 1987**) or their origin, respectively. After the calculation follows the matching chart with a layer on top, it contains the displayable graph in a transparent GIF- image. At the first time the layer is hidden and takes no space in the document.

### 7.2.2 JavaScript

The script consists of the four main parts. The first part checks, while loading the document, if the optimal browser is used and sends the browser to the correct page, if needed.

```
if(navigator.appName=="Netscape")
  {
   top.location.replace("lfl_nn.htm");
  };
```

In the 'if' statement the string "Netscape" will be compared against the property 'appName' and the method 'replace( )' changes the property 'location'.

Then begins a function, which will be executed if the user presses the 'calculate' button in the document. The function consists of the parts: value reading, checking, calculation and output. First the landing field length will be controlled with an 'if' statement if the range is between 0 m and 10000 m, and when the input is not correct, the user will be informed via an alert dialog box. Then the calculation interrupts with the 'return' statement:

```
if(v<=0||v>9999)
  {
   document.form1.slfl.value="";
   document.all.item("Layer1").style.display="none";
   alert("The landing field length is invalid.
              \n\n It should be between 1m and 10000m.");
   return;
  };
```

The same procedure is applied to the maximum lift coefficient. The permitted range for the value is 0 to 10. Then the program queries the selected mass ratio of the aircraft:

```
for(i=0;i<3;i++)
  if(document.form1.range[i].checked==true)
  z=document.form1.range[i].value;
if(z==-1){z=document.form1.cusrange.value};
```

In addition, the calculation starts. When the custom value of the mass ratio was not given correct by the user, the program interrupts by the 'if' statement. The main calculation is done with equation (1.0), the result is rounded to the nearest integer.

```
if(z>0&&z<=1)
   {
    w=eval((v*.107*w)/z);
    v=eval(Math.sqrt(v)*1.7);
    v=Math.round(v);
    w=Math.round(w);
```

After the calculation, the results will be write back to the form in the result text fields:

```
document.form1.vapp.value=v;
document.form1.winglo.value=w;
```

At the end of the calculate function, the position of the graph will be computed and displayed, using a linear equation of the first order. The 'if' statement checks, if the graph is inside the display-range of the chart:

```
if(w>=300&&w<=980)
        {
         y=eval(.39*w-537.91);
         document.all.item("Layer1").style.posLeft=Math.round(y);
         document.all.item("Layer1").style.display="inline";
         k=1;
        }
```

The third part of the script changes the source of the image tag, if the user presses the "start animation" button. The string with the filename of the animation is assigned to the document property 'images.src'.

```
function runa()
  {
   document.images.lflani.src="lafi.gif";
  };
```

Part four of the script saves the result from the calculation in a cookie on the hard disk. First is checked if the calculation was done:

```
function scookie()
  {
   if(w==-1)
    {
alert("Please  produce  first  a  result  by  clicking  on  the  calculate  but-
ton.");
     return;
    };
```

Then the user has to confirm, if the result is the correct for saving:

```
   check=confirm("Save this?\n\nwing loading = "+w+" kg/m²");
   if(check!=false)
```

In the 'if' statement now the result will be saved. For the control of the result on the hard disk, the value will be read from the cookie back to the program and compared with the value from the calculation. If these values are not equal, the cookies are deactivated in the preferences of the browser and the user gets a message about it:

```
    {
     document.cookie="a="+w+"b"+k;
     start=document.cookie.indexOf("a")+2;
     end=document.cookie.indexOf("b");
     ww=document.cookie.substring(start,end);
     if(ww!=w)
      {
       alert("Cookies are deactivated on this computer!\n\nThe result could
not be saved.");
       return;
      }
```

If the result could be saved, the saved value is displayed again and the browser jumps to the bottom of the page:

```
     else
      {
     alert("Wing loading = "+ww+" kg/m² saved.");
      };
     document.location="#bottom";
    };
  };
```

## 7.3 Takeoff Requirements

### 7.3.1 Page Layout

The regulations are formatted uniform via the CSS - file. The regulations follows an animation which explains the expressions used in this requirement. The main calculation is made according to the formula (1.1).

$$\frac{\frac{T_{TO}}{m_{MTO} \cdot g}}{\frac{m_{MTO}}{S_W}} = \frac{k_{TO}}{s_{TOFL} \cdot \frac{r}{r_0} \cdot c_{L,\max,L}} \tag{1.1}$$

The formula (1.1) is created as an imagemap, every character is linked with its explanation at the characters explanations page 'equ_char.htm'. At the bottom of the takeoff field length page is the matching chart with its layer containing the takeoff field line, hidden in the first place.

### 7.3.2 JavaScript

The script consists of four parts: checking all parameters, calculating, animation control and value saving. The checking is the browser control, explained in chapter 7.2.2, and the value control of the measures takeoff field length with its range 0m to 10000m and takeoff lift coefficient with the range 0 to 10 according to the following example - script part:

```
if(w<=0||w>10)
   {
    document.form1.clmaxto.value="";
    document.layers.Layer1.visibility="hide";
    alert("The lift coefficient is invalid.
              \n\nIt should be between 0.1 and 10.");
    return;
   };
```

After that, the calculation follows. The result is rounded to six digits behind the decimal point and written as output in the result text field in the calculation table.

```
  w=eval((Math.round(1000000*2.34/(v*w)))/1000000);
  document.form1.tttwwl.value=w;
```

The afterwards presentation of the graph is divided into four possible cases. The quantity of the calculated result decides about the display of one of three different graphs in the matching chart. Is the range of the value not in one of the three graphs, the method 'alert' shows an information to the user:

```
if(w<0.00025||w>0.000833)
   {
    k=0;
    document.layers.Layer1.visibility="hide";
    alert("The result is not in the diagram !");
   };
if(w>=0.00025&&w<=0.000366)
   {
    k=1;
    y=eval(-387931*w-213);
    z=eval(-284483*w+29);
    document.Layer1.document.pic.src="startline1.gif";
    document.layers.Layer1.moveTo(Math.round(y),Math.round(z));
    document.layers.Layer1.visibility="show";
   };
if(w>0.000366&&w<=0.000538)
   {
    k=2;
    y=eval(-348837*w-227);
    z=eval(-145349*w-1.7);
    document.Layer1.document.pic.src="startline2.gif";
    document.layers.Layer1.moveTo(Math.round(y),Math.round(z));
    document.layers.Layer1.visibility="show";
   };
if(w>0.000538&&w<=0.000833)
   {
    k=3;
    y=eval(-101695*w-360);
    z=eval(-101695*w-15);
    document.layers.Layer1.document.pic.src="startline3.gif";
    document.layers.Layer1.moveTo(Math.round(y),Math.round(z));
    document.layers.Layer1.visibility="show";
   };
```

The animation control assigns the string with the filename of the animation to the document property 'images.src' according to the following example:

```
function runa()
  {
   document.images.tflani.src="tofi.gif";
  };
```

At the last the function "scookie( )", described in chapter 7.2.2, saves the calculated value of the ratio takeoff thrust to weight over wing loading.

## 7.4 Second Segment Climb Requirements

### 7.4.1 Page Layout

After the regulations, set off with light background, a graphic describes the matters and definitions at the second segment as part of the takeoff maneuver. Then the equations explanation follows, it explains the origin of all values especially wing aspect ratio with a link to statistics of aspect ratios needed in the calculation at the bottom of the page. The aspect ratio statistics were taken from **Loftin 1980** and collected from **Jane's 1995**, **Jane's 1996** and **Jane's 1997**. The main calculation is done according formula (1.2) and (1.3):

$$c_D = c_{D,0} + D\, c_{D,\,flap} + D\, c_{D,slat} + D\, c_{D,gear} + \frac{c_L^2}{p \cdot A \cdot e} \tag{1.2}$$

$$\frac{T_{TO}}{m_{MTO} \cdot g} = \left(\frac{N}{N-1}\right) \cdot \left(\frac{1}{L/D} + \sin g\right) \tag{1.3}$$

### 7.4.2 JavaScript

The script consists of three parts: checking all parameters, calculating and value saving. The checking is the browser control, explained in chapter 7.2.2, and the value control of the measures takeoff lift coefficient (between 0 and 10), aspect ratio (in the range 1 to 50) and the engine number (2, 3 or 4) according to the following example - script part:

```
for(i=0;i<3;i++)if(document.form1.engine[i].checked==true)
             z=document.form1.engine[i].value;
if(z!=2&&z!=3&&z!=4)
   {
    document.all.item("Layer1").style.display="none";
    alert("Please select the\nnumber of engines !");
    return;
   };
```

In this script, the selected engine number is queried within an 'if' statement and in a 'for' loop. If the user had not selected an engine number, the script interrupts the running calculation with the 'return' statement and an 'alert' message. After the value control follows the calculation according to formulas (1.2) and (1.3), using the 'PI' property:

```
v=eval(v/1.44);
   if(v<1.4){v1=0.03};
   if(v>=1.4&&v<1.6){v1=0.04};
   if(v>=1.6){v1=0.05};
   ld=eval(v/(v1+(v*v/Math.PI/w/0.7)));
```

The second part of the calculation is done with the 'switch' statement:

```
switch (z)
     {
      case "2" :
        w=eval(2*(1/ld+0.024));
        break;
      case "3" :
        w=eval((3/2)*(1/ld+0.027));
        break;
      case "4" :
        w=eval((4/3)*(1/ld+0.03));
        break;
     };
```

The calculation ends with drawing the graph, its possible range is 0.15 to 0.4. Is the graph not in this range, the second branch of the 'if' statement after 'else' is executed:

```
if(w>=0.15&&w<=0.4)
     {
      y=eval(-666*w+59.2);
      document.all.item("Layer1").style.posTop=Math.round(y);
      document.all.item("Layer1").style.display="inline";
      k=1;
     }
  else
     {
      document.all.item("Layer1").style.display="none";
      alert("The result is not in the diagram !");
      k=0;
     };
```

At the end the function "scookie( )" offers the possibility to save the result takeoff thrust to weight in the document's cookie. This function is explained in chapter 7.2.2.

## 7.5 Missed Approach Climb Requirements

### 7.5.1 Page Layout

The regulations, presented with light background, follows the equations explanation, it explains the origin of aspect ratio and all other values. The aspect ratio statistics were taken from **Loftin 1980** and collected from **Jane's 1995**, **Jane's 1996** and **Jane's 1997**. The main calculation is done according to formula (1.2) and (1.4).

$$\frac{T_{TO}}{m_{MTO} \cdot g} = \left(\frac{N}{N-1}\right) \cdot \left(\frac{1}{L/D} + \sin g\right) \cdot \frac{m_{ML}}{m_{MTO}} \qquad (1.4)$$

### 7.5.2 JavaScript

The script consists of three parts: checking all parameters, calculating and value saving. The checking is the browser control, explained in chapter 7.2.2, and the value control of the measures landing lift coefficient (between 0 and 10), aspect ratio (in the range 1 to 50), the engine number (2, 3 or 4) and the ratio maximum landing mass to maximum takeoff mass (range 0 to 1). Further, the landing gear position is inquired, because of the different regulations FAR 25, where the landing gear has to be extended and JAR 25, where the landing gear is retracted:

```
for(i=0;i<2;i++)
    if(document.form1.reg[i].checked==true)zzz=document.form1.reg[i].value;
```

The first part of the calculation adds the drag coefficient at zero lift and the drag coefficient from the flaps, depending on the lift coefficient, to the landing gear drag coefficient. Its value is determined in the HTML tag 'input'.

```
if(v<1.4){v1=0.03};
if(v>=1.4&&v<1.6){v1=0.04};
if(v>=1.6){v1=0.05};
v1=eval(1*v1+1*zzz);
```

The further calculation and the saving function ("scookie( )") are similar to the explained script in chapter 7.4.2 .

## 7.6 Coincide of Parameters

### 7.6.1 Page Layout

At first the values and graphs from the requirements are presented, then it follows a text framed in a 'div' or 'layer' tag respectively and explains the determination of the matching point. Is one of the requirements from chapter 7.2 to 7.4 missing, a message is displayed instead and the sizing process cannot be finished graphically. The results from the matching chart can be compared with values from existing aircraft in the table below. Then follows explanation of the determination of the aircraft parameters maximum takeoff mass, operating empty mass, fuel mass, takeoff thrust and wing area and all formulas are shown. The mass fractions, needed in the fuel mass calculation, are explained at a different site, where the statistic determination with the help of **Roskam 1989** and the calculation with the Breguet factors including the loiter time from **FAR 125** is demonstrated. At the bottom of the page "matching chart" is at last the calculation table of the following aircraft parameters:

- maximum payload mass
- maximum takeoff mass
- operating empty mass
- fuel mass
- takeoff thrust
- wing area.

For information are also given the mass fractions for loiter and cruise. A second table with values from existing aircraft enables the comparing of all values.

### 7.6.2 JavaScript

The script begins with the collection of all saved values from the requirements landing field, takeoff field, second segment climb and missed approach. It writes, when the value was not saved a string to an error message in the variable "l" and presents it after the matching chart in the document. The following example shows the query of the landing requirement:

```
start=document.cookie.indexOf("a")+2;
end=document.cookie.indexOf("b");
k1=document.cookie.substring(end+1,end+2);
```

The 'if' statement contains the string for the error message, in the 'else' passage the is read from the cookie and its unit added:

```
if(k1!=1)
   {
    l=l+"<li>landing field length requirement</li>";
    ww=v;
   }
else
   {
    w=document.cookie.substring(start,end);
    ww=w+" kg/m&sup2;";
   };
```

The same procedure is applied to the other requirements. It follows a function 'paintlay', it will be executed when the whole document was loaded and with it, all layers are established. The parts of the function are: calculation of the graph position, painting the graph and changing the property 'display' in order to make the graph visible:

```
if(k1==1)
      {
       www=eval(.39*w-22);
       document.all.item("Layer1").style.posLeft=Math.round(www);
       document.all.item("Layer1").style.display="inline";
      };
```

If one of the requirements were not found in the cookie the prepared error message will be displayed, else the layers with the cruise graph and the match point - explanation:

```
if(k1!=1||(k2!=1&&k2!=2&&k2!=3)||k3!=1||k4!=1)
   {
    document.all.item("Layer6").style.display="inline";
   }
else
   {
    document.all.item("Layer5").style.display="inline";
    document.all.item("Layer7").style.display="inline";
   };
```

At last come two functions for calculation of all further aircraft parameters at the bottom of the site 'matching chart'. The first function calculates the maximum payload, the second maximum takeoff mass, operating empty mass, fuel mass, take off thrust and wing area. Both functions consist of value checking, calculating and output. For the maximum payload first the passenger mass is queried:

```
for(i=0;i<2;i++)
  if(document.form1.paxm[i].checked==true)
    paxm=document.form1.paxm[i].value;
```

Further the passenger number is queried and controlled with the range 0 to 1000, the cargo mass with its range 0kg to 100000kg:

```
var pax=document.form1.pax.value;
if(pax<0||pax>1000)
   {
    document.form1.pax.value="";
    alert("The passenger number is invalid.\n\nIt should be between 0 and
            999.");
    return;
   };
var cargo=document.form1.cargo.value;
if(cargo<0||cargo>100000)
   {
    document.form1.cargo.value="";
    alert("The cargo mass is invalid.\n\nIt should be positive or zero.");
    return;
   };
```

It follows the calculation and the output to the document:

```
numerator=eval(pax*paxm+1*cargo);
document.form1.mpl.value=Math.round(numerator);
```

The variable "numerator" is the numerator for the maximum takeoff mass calculation in the function "calcu2( )". In this function all other values checked with the following ranges:

- numerator: less or equal 0,
- maximum thrust to weight ratio: between 0 and 1;
- range: 1 to 100000 km
- Mach number: greater than 0,
- flight height: 0 to32 km,
- specific fuel consumption: greater than 0,

- wing loading: greater than 0,
- lift to drag ratio: greater than 0,
- the product of the marked fuel mass fractions: between 0 and 1.

The mass fractions are set to 1 when its checkbox was unmarked by the user:

```
var ff11=(document.form1.ff1.checked==true)? document.form1.ff11.value:1;
```

The calculation follows. The temperature is calculated depending on the height, between 0 and 11 km according to (1.5), between 11 km and 20 km equal to 216.65 Kelvin and from 20 km according to formula (1.6).

$$T = 288.15 - 6.5 \cdot H \tag{1.5}$$

$$T = 216.65 + (H - 20) \tag{1.6}$$

```
if(hgt<11){tp=eval(288.15-6.5*hgt)}
if(hgt>20){tp=eval(216.65-hgt+20)}
else{tp=216.65};
```

Then the velocity, the Breguet time factor and the cruise mass fraction with an addition of 370.4-km (200 N.M.) for the alternate airport requirement (**FAR 125**) are calculated:

```
var vel=eval(mach*20.05*Math.sqrt(tp));
var bt=eval(1E6*ld/sfc/9.81);
var cru=eval(Math.exp(-1000*(1*range+370.4)/bt/vel));
```

For the loiter mass fraction, needed with the Breguet range factor in the fuel mass calculation the loiter time has to be determined. In the short and middle range sector the loiter time is 45 min, in the long range sector 30 min plus 10 % of the flight time (**FAR 125**):

```
var loi,loi2;
if(range<5600)
   {
    loi=eval(Math.exp(-2700/bt))
   }
else
   {
    loi2=eval(1*1800+range*1000/vel);
    loi=eval(Math.exp(-loi2/bt))
   };
```

Then all further aircraft parameters, operating empty mass, fuel mass, take off thrust and wing area, are calculated. For the maximum takeoff mass is checked if the results will be convenient, by refuse the calculation when the denominator is less or equal zero:

```
var omass=eval(0.23+1.04*mttw);
var fmass=eval(1-ff*cru*loi);
var denominator=eval(1-(fmass)-(omass));
if(denominator<=0)
   {
    alert("The results are not convenient.");
    return;
   };
var mass=eval(numerator/denominator);
omass=eval(omass*mass);
fmass=eval(fmass*mass);
var tto=eval(mass*9.81*mttw/100);
var warea=eval(10*mass/mtos);
```

At last, the output is made with several roundings; the mass fractions are rounded to three digits behind the decimal point, the masses to whole numbers and the takeoff thrust and wing area to one digit behind the decimal point:

```
document.form1.loi.value=eval(Math.round(1000*loi)/1000);
document.form1.cru.value=eval(Math.round(1000*cru)/1000);
document.form1.mass.value=Math.round(mass);
document.form1.omass.value=Math.round(omass);
document.form1.fmass.value=Math.round(fmass);
document.form1.tto.value=eval(Math.round(tto)/10);
document.form1.warea.value=eval(Math.round(warea)/10);
```

# 8 Pros and Cons of Course Design with HTML and JavaScript

The most advantage is the plainness of the language HTML. No complicated software programs, their using and operations have to be learned. A simple text editor, such as Windows' Notepad is a sufficient program. The additional computer language JavaScript is compiled in runtime and written as plain text. Without any dependencies on software or manufacturers, HTML and JavaScript can be used by everyone and every time and are still useable and capable of development, when other document types have reached their limits. The text format of this document type has also the advantage that they are small documents, portable via Floppy disk, also quick and cheap for a transmission across the Internet. The document type HTML is not specific for a special trade; therefore this types are widespread and global used and well known by most people. It is applicable to all operating systems such as Windows and Unix.

JavaScript is based on the well tried and reliable computer language C. Several simplifications make it furthermore easier to use JavaScript in the engineering sector, where other languages, e.g. Fortran, are taught. Thus only the statement 'var' makes the variable declaration; complicated type converting can be dropped. JavaScript is not compiled through the programmer, its script can be controlled and inspected easy and every time from the browser. A multimedia course is extendable through the unlimited addition of further HTML sites and links.

Disadvantages are the dependencies on browser software. Because of the usage of newer functions, such as 'layer', are browser programs from newer generations needed. The inability of JavaScript to draw graphics is further a disadvantage in the multimedia course design. If these functions are indispensable needed, the programmer has to switch to computer languages like Java.

For the technical trade, the inability or difficulty to draw formulas can be handled with the embedding of pictures, where the formula preparation is done with external graphical programs. Further complex calculations with JavaScript are limited, because of the simplification of the variable definitions; integer, characters and floating point numbers are not separate declared; and the missing of compilation, the runtime of the scripts increase fast. In this sector, also the usage of Java or the like is required.

# 9 Summary

With its scripting language JavaScript, HTML is a possibility to present knowledge and information not only quick and easily accessible but also sophisticated in graphic and design. Layers arrange text and images dynamically, forms allow interactivity with the user and cookies can save results and settings. Calculations can be done while viewing the document, the scripting language JavaScript offers as much possibilities as a conventional programming language. The calling of a function – defined with the 'function' statement – from a form in the document is useful, it gives the chance to understand technical parameters and its effects by calculating parts of the design process repeatedly. In five steps, the multimedia course 'preliminary sizing in aircraft design' shows the combination and connections of important aircraft parameters, like the maximum lift coefficient or the aspect ratio. Its effects on the wing area and the maximum takeoff thrust, for instance, are understandable with several diagrams, produced with 'layer', 'ilayer' and 'div' tags. Cookies collect results from partial calculations. At the end of the course, these results will be compiled in the matching chart and with it, the user doesn't need paper to get final results.

Multimedia courses in HTML and JavaScript are practicable solutions in designing learning courses. They are easily extendable, platform independent and small and cheap for transmission via the Internet.

# Bibliographical Reference

| | |
|---|---|
| **Dubs 1987** | DUBS, F.: *Aerodynamik der reinen Unterschallströmung.* Basel: Birkhäuser, 1987 |
| **ECMA 1998** | ECMA: *Standardizing Information and Communication Systems.* 114 Rue du Rhône, CH-1204 Geneva Switzerland - URL: http://www.ecma.ch/ (09.12.99) |
| **FAR 25** | FEDERAL AVIATION ADMINISTRATION: *Airworthiness Standards: FAR Part 25: Transport Category Airplanes*. FAA, 1998 - URL: http://www.faa.gov/avr/ (15/10/98) |
| **FAR 125** | FEDERAL AVIATION ADMINISTRATION: *Airworthiness Standards: FAR Part 125: Certification and Operations: Airplanes having a seating capacity of 20 or more passengers...* . FAA, 1999 - URL: http://www.faa.gov/avr/ (28/10/99) |
| **Jane's 1995** | JACKSON, Paul (Ed.): *Jane's all the World's Aircraft 1995-96*. Coulsdon : Jane's Information Group Limited, 1995 |
| **Jane's 1996** | JACKSON, Paul (Ed.): *Jane's all the World's Aircraft 1996-97*. Coulsdon : Jane's Information Group Limited, 1996 |
| **Jane's 1997** | JACKSON, Paul (Ed.): *Jane's all the World's Aircraft 1997-98*. Coulsdon : Jane's Information Group Limited, 1997 |
| **JAR 25** | JOINT AVIATION AUTHORITIES: *Joint Aviation Requirements, JAR-25, Large Aeroplanes.*- Distribution: Civil Aviation Authority, Cheltenham, UK |
| **Loftin 1980** | LOFTIN , Laurence K., Jr.: *Subsonic Aircraft* : *Evolution and the Matching of Size to Performance* . NASA RP-1060, National Aeronautics and Space Administration Washington, DC, 1980 |
| **Microsoft 1999** | MICROSOFT CORPORATION: *JScript Language Reference.* Microsoft, 1999 - URL: http://msn.microsoft.com/tml/ (03/10/99). |

| | |
|---|---|
| **Netscape 1999** | NETSCAPE COMMUNICATIONS CORPORATION: *Client-Side JavaScript Reference v1.3*, Netscape, 1999 - URL: http://developer.netscape.com/library/documentation/ (03/10/99). |
| **RFC 0825** | POSTEL, J.: *Request for Comments on Requests for Comments.* RFC 825, Network Working Group, 1982 - URL: http://rfc.fh-koeln.de/rfc/html/rfc0825.html (06/12/99) |
| **RFC 1738** | BERNERS-LEE, T.; MASINTER, L.; et al.: *Uniform Resource Locators (URL).* RFC 1738, Network Working Group, 1994 - URL: http://rfc.fh-koeln.de/rfc/html/rfc1738.html (06/12/99) |
| **RFC 1945** | BERNERS-LEE, T.; MIT/LCS.; et al.*: Hypertext Transfer Protocol -- HTTP/1.0* . RFC 1945, Network Working Group, 1996 - URL: http://rfc.fh-koeln.de/rfc/html/rfc1945.html (09/12/99) |
| **RFC 2068** | FIELDING, R.; UC IRVINE; et al.*: Hypertext Transfer Protocol -- HTTP/1.1* . RFC 2068, Network Working Group, 1997 - URL: http://rfc.fh-koeln.de/rfc/html/rfc2068.html (09/12/99) |
| **Roskam 1989** | ROSKAM, J.: *Airplane Design.* Vol.1: *Preliminary Sizing of Airplanes.* Ottawa, Kansas, 1989. - Distribution: Analysis and Research Corporation, 120 East Ninth Street, Suite 2, Lawrence, Kansas, 66044, USA |
| **W3C 1997** | WORLD WIDE WEB CONSORTIUM: *HTML4.0 Specification.* Ed.: Dave Raggett; Arnaud Le Hors; Ian Jacobs; et.al., W3C, 1997 - URL: http://www.w3.org/TR/REC-html40/ (06/12/99) |