

Java Scripting

Java

Everyone talks about Java. Whether Java will considerably determine the future of the Internet or not, is undecided. However you can build with Java interesting functions into an HTML page. If you want to use Java programs, you do not need a special configuration of the virtual Servers. In this section we introduce you to Java and to JavaScript. We show with some examples, how you can build a Java applet into your own program. We also deal briefly with JavaScript.

Use Java

Our servers support client side JAVA, JAVA applets and can detect .class Libraries. Server side JAVA is not offered for security reasons. On the Internet you can find some meaningful and some not so meaningful Java applets for the free download, which can be built directly into you own HTML pages. Partly they make it possible for some very nice special effects, like graphic input surfaces, buttons, scrolling text, diagrams, and much more. Often you can find it as freeware, however it is desired to mention the developer. Almost all applets are available as source code.

If the program author provides the applet translated you do not need a Java program compiler for the application of this program.

However you should operate with Windows95 or WindowsNT, since normally long file names are used. Translated applet have the file extension .class. Often it is enough to build the applet with the tag <APPLET> into your HTML source code. In many cases it is necessary, to adapt special control files to the own desires and conditions. These control files are present normally in the ASCII format and are processed with a simple editor.

In most freeware applets, the source code is connected. On the one hand, this offers a good chance of learning the development of Java applets. On the other hand, the applets shape a basic structure around your own applets which you can develop faster or to complement your own ideas. Or simply only to see, what is possible with Java. However in order to build a finished applet (CLASS) into your homepage, the Java source code (" java" files) is not needed. This is an important difference of Java applets compared with Javascript programs. Java scripts are only included in the HTML page and are translated by the browser. Therefore everyone can read and re-use the code. Homepages, which use applets, consists sometimes of multiple different files.

An applet tag in a HTML page has the following basic structure:

```
<applet [Parameter] >  
<param [Variable] >  
alternative Text for a non-Java capable Browser  
</applet>
```

At first the definition of the applet tag takes place. A Java-capable browser knows that the definition of the applet follows. The browser will load and start the applet automatically. Apart from the layout specification, which determines the position and size of the applet, the file indicates the Java code.

- **code:** After the keyword " code " the Java-capable browser expects the name of the
- **file:** (URL) with the Java-Byte-Code (JavaApplet.class).
- **width:** Width of the applet in pixels.
- **height:** height of the applet in pixels.
- **codebase:** Optional specification of a base address for this applet. All not absolutely indicated URLs (" HTTP://...") refer to the directory indicated here.

- **vspace:** This value indicates, how much space you need on the left or the right of the applet.
- **hspace:** This value indicates how much space you need above and below the the applet. Both " vspace ", and " hspace " are considered only with the " align"-adjustment " left " or " right " .
- **align:** The Align-Parameter is also known by the other HTML tags (tag). It indicates, how the Browser should align applet to the page or the text. The attributes left, right, top, middle, bottom etc. correspond from the tag.
- **name:** Symbolic name of the applet.

Just like a C-program the applet can transfer the parameter (variable). The developer of the applet determines their name and meaning. Therefore there is a description necessary to each call parameter, to merge the applet into an HTML page. A " param" tag indicates the name of the call parameter as well as the desired value, on which this is to be set.

<param name=[name of the call parameter] value=[worth]>

- **name:** Name of variable (this could be in quotation marks)
- **value:** Value of variable. It is permitted to use number-value and string-value (should be with character strings and quotation marks).

Since web browsers ignore unknown tags, there is a simple possibility offered to produce an alternative text in a not Java-capable browser. If a text without tag or no <param> tag shows up within an applet definition, then the text is ignored by the Java browser. With a non Java-capable browser it works opposite, the unknown <applet> - and <param> tags are being ignored, so that only text is being displayed.

```
<applet code=LinkButton.class" width=50 height=30>
<param name=lbl value="MyLink">
<param name=href value="http://myserver/myhomepage.html">
<param name=sound value="sounds/click.au">
You need a java-capable browser
</applet>
```

In the example a Java browser would display a button with the inscription "MyLink". If you click on the button you hear the sound "click" and the page with the URL "http://myserver/myhomepage.html" is called. With a non Java-capable browser instead the text "You need a Java-capable browser " would appear. As usual </applet> terminates the <applet> block.

Application Examples with Finished Java Applet

The prepared Java applets are usually documented quite well. Normally an HTML file is available, in which the application applet is shown. You simply modify these files, this is the safest way to bring the example to run. If the applet needs help files, as for example Audio or Graphic files, then often the directory structure is given. That means, the directory structure from the respective example must be taken over. Pay attention also to upper and lower case! The specifications usually must be kept accurately.

Many example applets originate from JDK (Java Development Kit) of Sun (java.sun.com).

JavaScript - Programs in HTML Pages

JavaScript programs are transferred within an HTML document in the source text. A JavaScript-capable Browser has an interpreter, which translates and executes the program. JavaScript, a

community project of Sun and Netscape, is still not standard, but is still in the development stage. In the mean time the Netscape Browsers version 2.0 and higher and Internet Explorer version 3.0 or higher can process Java script programs. JavaScript has like Java borders. As it is not meaningful to implement in Java a whole page layout, you should also not write complex programs in JavaScript.

JavaScript is suited for smaller functions, like the output of text in the status line of the Browser-window, communication between objects and frames or the examination of user inputs (if a correct E-Mail address is indicated etc.).

In this paragraph you find a first overview of the concepts of JavaScript.

The target is to understand the basic structure of a JavaScript program and to incorporate finished JavaScript programs in your own site. If after reading this section, you would like to know more about JavaScript, then we refer to the countless books about this topic. You can also find information about it on the Internet. At Netscape's web site (<http://www.netscape.home>) you can always find the complete and current JavaScript description.

The name JavaScript suggests a relationship between the Script language and the object-oriented programming language Java. Despite many common things both differ in substantial points from each other.

JavaScript Java

JavaScript is interpreted by the WWW Browser on the Client computer. The program source text is not compiled. Java applet are compiled in byte code. This is then transferred and processed by the Java Virtual Engine.

JavaScript is object-based, however it does not permit class definition or transmission. The browser administers the objects. Java is an object-oriented programming language. The typical features such as classes, objects, transmission etc. are fully available. " dynamic binding " - object references are only resolved at run-time. If a function is used, which was not defined, the Browser announces only the error after it is actually called. " static binding " - objects are referenced when compiling, i.e. all references are well-known at run-time. If a reference cannot be made, then the error occurs when compiling.

Variables do not have to be defined as in Java, C or Pascal. Variables must be defined before use.

Slower than Java, since the program is only interpreted at run-time.

Faster than JavaScript, since Java applet are already completely compiled.

The source text of a JavaScript program is contained fully in the HTML page The HTML page contains only the call of an applet. The source code is not needed after compiling.

Hello World

There are two possibilities to merge a JavaScript program into an HTML page: Either you use the new imported <SCRIPT> tag or you operate with " event handlers ". After a <SCRIPT> tag, follows a normal program with instruction and function. Event-Handler on the other hand is connected with an HTML object and reacts to user inputs. They generally call a <SCRIPT> section defined function, which for example checks the inputs into a text field.

As the first well-known " Hello World" demo cannot be missing. A JavaScript program always begins with the <SCRIPT> tag and ends with </SCRIPT>:

```
<SCRIPT>
```

```
... JavaScript-Program
</SCRIPT>
```

Optional kann das <SCRIPT>-Tag auch noch durch die Angabe LANGUAGE" ergänzt werden.
Optionally a <SCRIPT>-tag can also be used " LANGUAGE".

```
<SCRIPT LANGUAGE="JavaScript">
... </SCRIPT>
```

The following short program shows the basic structure of a JavaScript program. The JavaScript should be located basically in the " HEAD" section of the HTML page. Since this is processed first while loading the page, it is guaranteed that the Script is entered and processed, before the user can initiate an action. The " Comment" tag in the SCRIPT block is noticeable.

This has the function, to hide the Script with an old browser (" Code-Hiding "). A JavaScript-capable browser ignores the comment tag within a SCRIPT container and processes the script completely. An old browser, which does not know JavaScript, treats the entire block as a comment and ignores it. Without this "Comment" tag the old browser would make the JavaScript text on the HTML page visible.

Note: *The closing comment tag must be located in one line, which begins for its part with a JavaScript comment symbol (//), otherwise the Interpreter will announce a syntax error.*

```
<HTML>
<HEAD>
<SCRIPT>
<!-- begin the script ...
//the "COMMENT" tag hides the script for old browsers
// JavaScript
document.write ("my first JavaScript Program")
// End the Script.. COMMENT-tag closed -->
</SCRIPT>
</HEAD>
<BODY>
// contents of the web page
</BODY>
</HTML>
```

The browser produces automatic objects, which are available in JavaScript. Thereby the appropriate methods and data fields (attributes, characteristics) can be used. As soon as an HTML page is loaded, there is for example the object " document ". This is for the current HTML windows on the display.

The object has the method " write() ", which outputs a text in the window. The call " document.write (" my first JavaScript program ") outputs therefore the text " my first JavaScript program ". Try the example out. Despite the extensive HTML page the screen only displays the text " my first JavaScript program ".

Note: *The print function of the Browsers ignores such Java generated texts.*

JavaScript is " case sensitive ", i.e. it differentiates between upper and lower case. Therefore you must always pay attention to the way of writing the objects and methods. The method " LastModified " of the object " document " supplies for example time-of-day and date of the last modification of the HTML file. The call " document.lastmodified " supplies an error. Directions which are located directly in the Script Block, are going to be loaded and processed automatically by the browser. In addition, JavaScript enables the definition of functions. A function is initiated with the keyword " function ". Afterwards the function name as well as the call parameters will

follow. A Java or C-programmer will notice immediately that no type declarations are made. The function returns an integer value in the example, yet this does not have to be indicated at the beginning.

```
<HTML>
<HEAD>
<TITLE>jcr01.htm: First JavaScript-Demo</TITLE>
<SCRIPT>
<!-- begin the Script ...
//the "COMMENT"-tag hides the Script for old Browsers
function output number (n)
{
document.write ("function output number (", n, "): ")
for (i = 0; i < n; ++i)
{
document.write (i)
}
document.writeln ("<p>")
return (n)
}
document.writeln ("Result of the function call: ", Output number (10), "<p>")

// End the Script.. comment-tag closed -->
</SCRIPT>
</HEAD>
<BODY>
Here is the HTML-BODY
</BODY>
</HTML>
```

The sample program supplies the following output on the display when the loading in Netscape:

```
function output number (10): 0123456789
Result of the function call: 10
Here is the BODY
```

As expected the JavaScript which is inserted in the Head block is processed first and then the content of the HTML page (Body area). The method " write() " by the way considers the output of the usual HTML tags. This applies to all output functions with JavaScript. To move a line therefore you need to use <p>.

[Home](#) | [HTML I](#) | [HTML II](#) | [Graphics](#) | [FTP](#)
[Hosting](#) | [Scripts](#) | [Java](#)