```
                        // und Daten zum Attribut laut Sicherung lesen
                        var InputDaten = InputDatenObjekt.getAttribute(FreiesCacheAttribut);
                }

        </SCRIPT>
        </HEAD>
        <BODY>
                <FORM ID="ID_Formular">
                        <INPUT   ID="ID_Input"
                                CLASS="user_data_speicher_klasse"
                                TYPE="text"
                        >
                        <INPUT TYPE="button" VALUE="sichern der Input-Daten" onclick="InputSichern()">
                        <INPUT TYPE="button" VALUE="laden der Input-Daten"    onclick="InputLaden()">
                </FORM>
        </BODY>
        </HTML>
```

Eigenschaften:

.expires — Zeitstempel der Daten im UserDataStore (User-Cache) per .style.userData Behavior
Zeitstempel als Verfallszeitpunkt für automatische Löschung gecachter Daten:
Daten werden **automatisch** durch den Browser gelöscht, wenn das Datum der Daten anhand des Zeitstempels als verfallen erkannt wurde, also der aktuelle Zeitpunkt laut PC-Uhr jünger ist, als der Zeitstempel.
String im UTC (Universal Time Coordinate)-Format

Methoden:

.load() — gespeicherten UserDataStore (User-Cache) auslesen per .style.userData Behavior
Cache muss per Methode .save() zum Behavior gespeichert worden sein

.save() — UserDataStore (User-Cache) speichern per .style.userData Behavior

4.3.2.2.4.3.39.25. HTML-Dokument mit Style-Sheet (CSS) im IE und NS

Nachfolgende Beschreibung ist die **HTML-konforme** Erweiterung des Objekt-Eigenschaft .style.

Aufgrund der HTML-Integration von CSS sollte der Browser CSS unterstützen. Ob diese Unterstützung vollständig ist, hängt vom Willen des Browserherstellers und seiner am Markt angebotenen Browserversionen ab. Dabei gilt: Auch wenn der HTML-Standard Style-Sheets implementiert hat und ab HTML 4.01 diverse Tags und Attribute als deprecated (missbilligt) setzt, weil sie vereinheitlicht durch genormte Stylesheets zu ersetzen sind, kann der Browser CSS nur im Umfang der implementierten Objekt-Eigenschaften von .style unterstützen. Wie immer gilt: Probieren geht über Studieren.

Microsoft setzt verstärkt auf XML, dessen Komponenten auf Basis der browsereigenen Objektimplementation mehr oder weniger HTML und CSS unterstützen. XML ist die perfektere Abbildung der browser-internen Objektstruktur und somit eine intern komplett objektorientierte Scriptsprache, allerdings hersteller- und browserspezifisch. Letztendlich wird CSS sinnigerweise als grundlegende Komponente und Eigenschaft **aller** Objekte integriert. Ein Vorteil, den Script-Programmierer beim Internet Explorer zu schätzen (lernen) wissen und den andere Browserhersteller mehr und mehr ihren Usern anbieten. CSS und .style bieten elementare Vielfalt bei der dynamischen Scriptprogrammierung, sind gewöhnungsbedürftig aber durch ihre Normung universell anwendbar.

Es sei darauf hingewiesen, dass die Verwendung von CSS-Klassen, Kodierung von <STYLE> bzw. dem STYLE-Attribut bzw. der Eigenschaften.style alle synonym sind. Vier Formen für dieselbe Sache, aber mit folgenden Unterschieden: CSS-Klassen können dokumentweit verwendet werden; <STYLE>, STYLE-Attribut und .style sind HTML-komponentenbezogen.

Hinweis: Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
Padding: Abstand des Objektinhaltes zum Aussenrand

Bsp: zu .style: Es sollen DIVs erzeugt werden und deren Objektreferenzen (Zeiger) in einem Zeigerfeld gesammelt werden. Das Zeigerfeld dient der vereinfachten Verwaltung der dynamisch erzeugten DIV's.

```
var DIV_ID="Test_DIV"; //  auch "Otto_DIV" etc.möglich , je nach Wunsch

var DIV_Zeiger=Array(); // sammelt alle ID als Zeichenketten

// DIV's dynamisch erzeugen und Zeiger einsammeln
var Left= 20;           // Abstand vom linken Fensterrand in Pixeln
for (var i=0; i < 3; i++)
{
        Left+=10; //  ab 30 Pixel vom linken Fensterrand
        document.write(         '<DIV ID="' + DIV_ID + i.toString() + ""
                        +       '  STYLE="position:absolute;'
                        +                   'left=' + Left + 'px;'
                        +                     'top=20px;'
                        +                     ""
                        +  '>'
                        +  '</DIV>'
                      );
        eval ('DIV_Zeiger[' + i + '] =' +  DIV_ID + i.toString() );
}
```


```
// DIV's dynamisch auf dem Bildschirm bezüglich dem linken Fensterrand um 20 Pixel
//                          verschieben
Left= 40;
for (i=0; i < 3; i++)
{
        Left+=10; // ab 50 Pixel vom linken Fensterrand verschieben
        eval('document.all.DIV_Zeiger[' + i + '] .style.left=' + Left);
}

Hinweis: Anstelle von i.toString() kann auch nur  i  kodiert werden, da der Browser
                    aufgrund von document.write()  i  automatisch nach String umwandelt.
        analog für i.toString() innerhalb eval()

Die Verschiebung erfolgt natürlich mit den Objekten, welche innerhalb von <DIV> .. </DIV>
        kodiert wurden. Im Falle von IMG's innerhalb des DIV werden diese mit verschoben.

Vorallem wegen der Manipulation von  HTML-Objekten werden DIV's verwendet.
```

4.3.2.2.4.3.39.25.1. Style-Sheet-Deklarations-Varianten

Werden Eigenschaften innerhalb von Tags kodiert, so gilt:
Sind Tags verschachtelt, so werden die Eigenschaften vererbt
--> Überschreiben innerhalb der Erb-Ebene möglich

Kommentierung nur innerhalb 1 Zeile per
nicht jeder Browser unterstützt Style-Sheet, also Style-Sheets innerhalb von <!-- //--> kodieren

für Tags wird nicht unterschieden zwischen Groß und Klein

4.3.2.2.4.3.39.25.1.1. Style-Sheet Eigenschaften innerhalb <HEAD> deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        @eigenschaften                          /* festkodiert ist @
                //-->
        </STYLE>
</HEAD>
<BODY>
....
</BODY>
```

Bsp: @import

4.3.2.2.4.3.39.25.1.2. Style-Sheet-Format und Style-Sheet-Eigenschaften als tag-unabhängig deklarieren (Attribut ID)

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        #format_name{eigenschaften_liste}   /* festkodiert ist #
                //-->
        </STYLE>
</HEAD>
<BODY>
        <tag_name ID="format_name">  ..... </tag_name>
</BODY>
```

format_name: Bsp: fettkursiv
innerhalb STYLE mit vorgesetztem # kodieren !
innerhalb BODY ohne # kodieren !
eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

Hinweise: ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das
tag-unabhängige ersetzt.
tab-unabhängige Style-Sheet-Format fügen sich ansonsten zu den tag-abhängigen hinzu.

4.3.2.2.4.3.39.25.1.3. Style-Sheet-Schlüsselwort und Style-Sheet-Eigenschaften als tag-abhängig deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        tag_name:schlüsselwort{eigenschaften_liste}   /* Doppelpunkt ist festkodiert
                //-->
```


```
        </STYLE>
</HEAD>
<BODY>
        <tag_name>  ..... </tag_name>
</BODY>
```

tag_name:schlüsselwort
Bsp: a:link

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

Hinweise: ein gleichnamiges tag-abhängiges Style-Sheet-Format wird in der Darstellung durch das tag-unabhängige ersetzt.
tab-unabhängige Style-Sheet-Format fügen sich ansonsten zu den tag-abhängigen hinzu.

4.3.2.2.4.3.39.25.1.4. Style-Sheet Eigenschaften ohne Tag-Attribut ID deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        tag_namen_liste{eigenschaften_liste}
                //-->
        </STYLE>
</HEAD>
<BODY>
</BODY>
```

tag_namen_liste: Kommatrennung
alle Tags haben genau eine gemeinsame Eigenschaftsliste
Bsp: h1,h2

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.1.5. Style-Sheet-Unterklasse deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        tag_name.unterklasse_name{eigenschaften_liste}      /* Punkt ist festkodiert
                oder    all.unterklasse_name{eigenschaften_liste}           /* Punkt  und all sind festkodiert
                oder    .unterklasse_name{eigenschaften_liste}              /* Punkt ist festkodiert
                //-->
        </STYLE>
</HEAD>
<BODY>
        <tag_name CLASS="unterklasse_name"> ..... </tag_name>
</BODY>
```

tag_namen.unterklasse_name: Bsp: p.normal
all.unterklasse_name: Unterklasse gilt für ALLE Tags
.unterklasse_name: identisch mit **all.**unterklasse_name

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.1.6. Style-Sheet-Eigenschaften mit Attribut STYLE deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--

                //-->
        </STYLE>
</HEAD>
<BODY>
        <tag_name STYLE=eigenschaften_liste>..... </tag_name>
</BODY>
```


Bsp: <P STYLE= ....> .... </P>

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.1.7. Style-Sheet-Eigenschaften eines HTML-Elementes deklarieren

```
<HEAD>
        <TITLE>.... </TITLE>
        <STYLE TYPE="text/css">
                <!--

                //-->
        </STYLE>
</HEAD>
<BODY>
        <tag_name ..>
                <SPAN STYLE=eigenschaften_liste>
                        zu formatierendes HTML-Element
                </SPAN>
        ....
        </tag_name>
</BODY>
```

Bsp: <P STYLE= ....> .... </P>

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

HTML-Element MUSS Endetag besitzen, das kodiert werden MUSS
Bsp: <P STYLE="font-size:12pt;"> ..... </P>

4.3.2.2.4.3.39.25.1.8. Beispiele für Style-Sheet

Beispiel 1

```
<HEAD>
        <TITLE> ..... </TITLE>
        <STYLE TYPE="text/css">
                <!--
                        body{   background-color:#FFFFFF;
                                background-image:url(test.jpg);
                                background-repeat:repeat-y;
                                margin-top: 1cm;
                                margin-left: 2cm;
                                margin-right: 1cm;
                            }
                        p{      font-family:serif;
                                font-size: 12pt;
                                text-align:justify;
                          }
                        p.zusatz_format_p_tag{text-ident:0.5cm;}
                        .zusatz_format_alle_tags{
                                position:absolute;
                                top:4cm;
                                z-index:3;
                                font-size:40pt;
                                font-width:bold;
                                color:blue;
                                text-align:center;}
                -->
        </STYLE>
</HEAD>

<BODY>
        <SPAN CLASS=" zusatz_format_alle_tags"> text </SPAN>
        <P CLASS=" zusatz_format_p_tag" text </P>
</BODY>
```

<!-- --> für Browser kodieren, die CSS nicht können
background-image: Hintergrundbild test.jpg verwenden
Achtung: Möglichst absolute Urls bzw. Pfadangaben verwenden, da
eventuell Browser relative Angaben nicht interpretieren

		kann	
		Bsp.:	relativ :url('ordner/test.gif')
			absolut :url('www.test.de/ordner/test.gif')
background-repeat:	repeat-y	Hintergrundbild senkrecht wiederholen	
	repeat-x	Hintergrundbild waagerecht wiederholen	
	no-repeat	Hintergrundbild nicht wiederholen	
p{....}	gilt für JEDES P-Tag ohne CLASS-Attribut zu kodieren		
margin-xxxx	Seitenrand		
text-align	Textausrichtung		
text-ident	Einzug links		
position	Art der position-Angabe folgenden Angaben		
	Bsp:	position:absolute	--> Absolutangaben
		top:4cm	hier 4 cm
top	Abstand zum Browserfenster oben		
z-index	Angezeigte Schicht (analog zu LAYER)		
		--> ab 1 = unterste Schicht	
	verwenden für überlappende Textbereiche		
p.zusatz_format_p_tag{ ..}	per CLASS-Attribut anwenden im P-Tag		

Beispiel 2

```
<HTML>
<HEAD>
<TITLE> ..... </TITLE>
<STYLE TYPE="text/css">
<!--
        .eigene_css_klasse{ .........}
-->
</STYLE>
</HEAD>
<BODY>
<DIV CLASS=eigene_css_klasse>.....</DIV>
.....
</BODY>
</HTML>
```

4.3.2.2.4.3.39.25.2. Style-Sheet und numerische (nicht vordefinierte) Farbangaben

#rrggbb oder rgb(rrr,ggg,bbb) oder vordefinierte Farben

Rot-Anteil:		
	hexa	rr
	dezimal 0-255	rrr
Grün-Anteil:		
	hexa	gg
	dezimal 0-255	ggg
Blau-Anteil:		
	hexa	bb
	dezimal 0-255	bbb

4.3.2.2.4.3.39.25.3. Style-Sheet und vordefinierte Bezeichner

Die Nachfolgende Beschreibung umfasst nur die wichtigsten Style-Sheet-Eigenschaften.

4.3.2.2.4.3.39.25.3.1. Style-Sheet und vordefinierte Dimensionen

pt	Point	=	1/72 inches
pc	Pica	=	12 pt
in	Inch	=	2,54 cm
mm			
cm			
px	Pixel		
em	relativ zur Schrifthöhe des Elementes, das per CSS formatiert wird		
ex	relativ zur Höhe des Buchstaben Grosses X		
%	Prozentanteil von der Norm des per CSS formatierten Elementes		

4.3.2.2.4.3.39.25.3.2. Style-Sheet- und Dezimalkomma

Dezimalkomma ist der Punkt und NICHT das Komma !!!

4.3.2.2.4.3.39.25.3.3. Style-Sheet und vordefinierte Schriften (Font und Style)

vordefinierte Schriftarten:	serif	
	san-serif	
	cursive	
	fantasy	
	monospace	
vordefinierte Style:	italic	kursiv
	oblique	kursiv
	normal	nicht kursiv

4.3.2.2.4.3.39.25.3.4. Style-Sheet und vordefinierte Farbbereiche

4.3.2.2.4.3.39.25.3.4.1. Style-Sheet und Farbe des Desktop

background	Farbe	Hintergrund Desktop

4.3.2.2.4.3.39.25.3.4.2. Style-Sheet und Farbe des Dokumentfensters

window	Farbe Hintergrund Dokumentfenster
windowframe	Farbe Rahmen Dokumentfenster
windowtext	Farbe Text im Dokumentfenster

4.3.2.2.4.3.39.25.3.4.3. Style-Sheet und Farbe aktives Fenster

activeborder	Farbe aktiven Fenstertitelzeile (-leiste) oben
activecaption	Farbe Überschrift in der aktiven Fenstertitelzeile
appworkspace	Farbe Hintergrund aktives Fenster

4.3.2.2.4.3.39.25.3.4.4. Style-Sheet und Farbe inaktives Fenster

inactiveborder	Farbe	nichtaktive Fenstertitelzeile
inactivecaption	Farbe	Überschrift nichtaktive Fenstertitelzeile

4.3.2.2.4.3.39.25.3.4.5. Style-Sheet und Farbe des Dialogfensters

captiontext	Farbe	Überschrift im Dialogfenster
greytext	Farbe	deaktivierter Text im Dialogfenster
buttonface	Farbe	Button im Dialogfenster
buttonhighlight	Farbe	3D-Schatten von Button im Dialogfenster
buttontext	Farbe	Button-Text im Dialogfenster

4.3.2.2.4.3.39.25.3.4.6. Style-Sheet und Farbe einer Auswahlliste

highlight	Farbe	ausgewählter Eintrag in Auswahlliste
highlighttext	Farbe	selektierter Text in Auswahlliste

4.3.2.2.4.3.39.25.3.4.7. Style-Sheet und Farbe von Tooltip und Popuphilfe (Hint)

infobackground	Farbe	Tooltips und Popuphilfen (Hints)
infotext	Farbe	Text von Tooltips und Popuphilfen (Hints)

4.3.2.2.4.3.39.25.3.4.8. Style-Sheet und Farbe eines Menü

menu	Farbe	Menüleiste
menutext	Farbe	Menüeintrag

4.3.2.2.4.3.39.25.3.4.9. Style-Sheet und Farbe der Scrolleiste

scrollbar	Farbe	Scrolleiste

4.3.2.2.4.3.39.25.3.4.10. Style-Sheet und Farbe eines 3D-Elements

threeddarkshadow	Farbanteil dunkel für Schatten bei 3D-Element
threedface	Farbe 3D-Element
threedhighlight	Farbe gerade angeklicktes 3D-Element
threedlightshadow	Farbanteil hell für Schatten bei 3D-Element
threedshadow	Farbanteil dunkel für Schatten bei 3D-Element

4.3.2.2.4.3.39.25.4. Style-Sheet-Dateien

4.3.2.2.4.3.39.25.4.1. Style-Sheet-Schriftdatei laden (.eot bzw. *.prf)*

Microsoft: *.eot
Netscape *.pfr

```
<STYLE TYPE="text/css">
        <!--
                @font-face{font-family:familien_name;
                        src: url(eot_bzw._prf_datei_liste);}
                font-size: ...
                ...
        // -->
</STYLE>
```

4.3.2.2.4.3.39.25.4.2. Style-Sheet-Informationen aus Datei einbinden (.css)*

Variante 1:

```
        <LINK
                REL=stylesheet
                TYPE="text/css"
                MEDIA="typ_name"
                HREF="url_oder_css_dateiname"
        >
        <STYLE TYPE="text/css">
                <!-- weitere Stylesheet-Angaben
                // -->
        </STYLE>
```

typ_name:	Name des Ausgabe-Mediums	
	"all"	alle Medien
	"screen"	Bildschirm

"print" Drucker

Variante 2:

```
<LINK
        REL=stylesheet
        TYPE="text/css"
        [MEDIA="typ_name"]                  optional
        HREF="url_oder_css_dateiname"
>

<STYLE TYPE="text/css">
        <!--
                @import (url_oder_css_datei) typ_liste;
                ....
        // -->
</STYLE>
```

typ_liste: Liste der Ausgabe-Medien
Kommatrennung
Bsp: print,screen;

Beispiel für Seitenelemente vom Druck ausschließen (ab IE 4.x und NS 6.x):

Seitenteile, die nicht gedruckt werden sollen, in <DIV CLASS="keindruck"> und </DIV>
oder in <SPAN CLASS="keindruck"> und </SPAN> einschliessen.

zwischen <HEAD> und </HEAD> einfügen:
<LINK REL="stylesheet" MEDIA="print" HREF="print.css">

print.css enthält nur
.keindruck {display:none;}

Beispiel für Seitenelemente nur beim Druck anzeigen (ab IE 4.x):

Seitenteile, die nur auf Ausdrucken sichtbar sein solllen, in <DIV CLASS="nurdruck"> und </DIV>
oder in <SPAN CLASS ="nurdruck"> und </SPAN> einschliessen

zwischen <HEAD> und </HEAD> einfügen:
<LINK REL ="stylesheet" MEDIA="screen" HREF="screen.css">

screen.css enthält nur
.nurdruck {display:none;}

4.3.2.2.4.3.39.25.4.3. Style-Sheet und Ausgabemedien anhand browserinterner Pseudoklassen

4.3.2.2.4.3.39.25.4.3.1. Ausgabemedien-spezifische CSS-Datei importieren (@import)

@import **(**css_datei**)** typ_liste**;**

typ_liste: Liste der Ausgabe-Medien
Kommatrennung
Bsp: print, screen;

4.3.2.2.4.3.39.25.4.3.2. Ausgabenmedien-spezifische Style-Sheet-Eigenschaften deklarieren (@media)

@media typ_liste**{**eigenschaften_liste**}**

typ_liste: Liste der Ausgabe-Medien
Kommatrennung
Bsp: print, screen;
eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.4.4. Style-Sheet und Font-Dateien (@font-face) anhand browserinterner Pseudoklasse

@font-face **{**eigenschaften_liste**;**
src: url **(**url_liste_mit_kommatrennung**);**
und/oder local **(**datei_namen_liste_mit_blank_trennung**);**
und/oder format (TrueType);
}

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.4.5. Style-Sheet und Seiten-Eigenschaften (@page) anhand browserinterner Pseudoklasse

@page:first {eigenschaften_liste}	Regel gehört der 1. Seite
@page:footer{eigenschaften_liste}	Regel gehört der Fußnote
@page:header{eigenschaften_liste}	Regel gehört der Kopfnote
@page:left{eigenschaften_liste}	Regel gehört der linken Seite
@page:left : header{eigenschaften_liste}	Regel gehört der Kopfnote Seite links
@page:left : footer{eigenschaften_liste}	Regel gehört der Fußnote Seite links
@page:right{eigenschaften_liste}	Regel gehört der rechten Seite
@page:right:header{eigenschaften_liste}	Regel gehört der Kopfnote Seite rechts
@page:right:footer{eigenschaften_liste}	Regel gehört der Fußnote Seite rechts

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen

Beispiel: @page : left {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.4.6. Style-Sheet und Hintergrund-Grafik

background-image: url(grafik_datei**);**			Hintergrunddatei GIF oder JPG
background-color:#rrggbb; oder rgb(rrr,ggg,bbb);			Hintergrundfarbe
background-repeat:	repeat;		Hintergrundgrafik kacheln auf gesamten Hintergrund
	oder	repeat-x;	Hintergrundgrafik für 1 Zeile kacheln
	oder	repeat-y;	Hintergrundgrafik für 1 Spalte kacheln
	oder	no-repeat;	kein Kacheln des Hintergrundbildes
background-attachment: scroll;			Hintergrundgrafik scrollt mit Vordergrund
	oder	fixed;	Hintergrundgrafik scrollt nie
background-position: top;			Hintergrundgrafik auf Oberkante Hintergrund
	oder	center;	Hintergrundgrafik zentriert auf Hintergrund
	oder	middle;	Hintergrundgrafik mittig auf Hintergrund
	oder	bottom;	Hintergrundgrafik auf Unterkante Hintergrund
	oder	left;	Hintergrundgrafik linksbündig auf Hintergrund
	oder	right;	Hintergrundgrafik rechtsbündig auf Hintergrund

background: eigenschaften_liste_mit_blank_trennung; obige Eigenschaften als Liste
Bsp: background: url(back.gif) no-repeat middle; Blank-Trennung !!

-moz-opacity:faktor — Transparenz des Hintergrundbildes
nur NS 6.x
Faktor > 0 und bis 1
Bsp: 0.9 entpricht 90%
Bsp: <IMG STYLE="-moz-opacity:0.9" SRC="test.jpg">

4.3.2.2.4.3.39.25.4.7. Style-Sheet und Rahmen-Eigenschaften (Border)

border-top-width: rahmen_dicke_von_rahmen_oberhalb_des_elementes;
border-bottom-width: rahmen_dicke_von_rahmen_unterhalb_des_elementes;
border-left-width: rahmen_dicke_von_rahmen_links_des_elementes;
border-right-width: rahmen_dicke_von_rahmen_rechts_des_elementes;

border-color:#rrggbb; oder rgb(rrr,ggg,bbb); Rahmenfarbe

border-style:	none;	kein Rahmen um das Element
oder	dotted;	gepunkteter Rahmen um das Element
oder	dashed;	gestrichelter Rahmen um das Element
oder	solid;	einfacher durchgezogener Rahmen um das Element
oder	double;	doppelter durchgezogener Rahmen um das Element
oder	groove;	3D-Effekt 1
oder	ridge;	3D-Effekt 2
oder	inset;	3D-Effekt 3
oder	outset;	3D-Effekt 4

border-top: eigenschaften_liste_1;
border-bottom: eigenschaften_liste_2;
border-right: eigenschaften_liste_3;
border-left: eigenschaften_liste_4;
border-width: eingenschaften_liste_5;

border:eigenschaften_liste_6;

Hinweise:

rahmen_dicke_.... : auch thin oder medium oder thick

eigenschaften_liste_1: wert_von_border-top-width #rrggbb wert_von_border-style
Werte mit Blank trennen

eigenschaften_liste_2: wert_von_border-bottom-width #rrggbb wert_von_border-style
Werte mit Blank trennen

eigenschaften_liste_3: wert_von_border-left-width #rrggbb wert_von_border-style
Werte mit Blank trennen

eigenschaften_liste_4: wert_von_border-right-width #rrggbb wert_von_border-style
Werte mit Blank trennen

eigenschaften_liste_5: Werte-Liste aller möglichen Bordereigenschaften
Werte mit Blank trennen

Anstelle von Doppelpunkt ist auch das Gleichheitszeichen kodierbar

Bsp: border-top: thick inset rgb(192,192,255);

4.3.2.2.4.3.39.25.4.8. Style-Sheet und HTML-Tag-bezogene Eigenschaften

eigenschaften_liste: Semikolontrennung
muss mit Semikolon enden
wenn Blank enthalten, so Liste in " " setzen
Bsp: {font-weight:bold;font_style:italic;}

4.3.2.2.4.3.39.25.4.8.1. Style-Sheet-Eigenschaften zu <A>

a:link{eigenschaften_liste} Eigenschaften für noch nicht besuchte Links
a:visited {eigenschaften_liste} Eigenschaften für schon besuchte Links
a:active{eigenschaften_liste} Eigenschaften für gerade besuchtes Link
a:hover{eigenschaften_liste} Eigenschaften für Maus über Link , ab IE 4.x
nur für Tags mit HREF-Attribut

4.3.2.2.4.3.39.25.4.8.2. Style-Sheet-Eigenschaften zu <P>

p:first-line{eigenschaften_liste} Eigenschaften Absatz 1. Zeile
p:first-letter{eigenschaften_liste} Eigenschaften Absatz 1. Zeile, 1. Zeichen
p:before{content:"freier_text" } Text vor dem Element einfügen
p:after{content:"freier_text" } Text nach dem Element einfügen

4.3.2.2.4.3.39.25.4.8.3. Style-Sheet-Eigenschaften zu <H1> bis <H6>

h1:first-line{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile
h1:first-letter{eigenschaften_liste} Eigenschaften Überschrift 1. Zeile, 1. Zeichen

für H2 bis H6 analog

Beim Internet Explorer 6.0 unter Windows 98 bzw. Windows XP wurde die Darstellung der Überschriften verändert, wenn der jeweilige Standardfont benutzt wird. Empfehlung: Neudefinition der <Hx>-Tags per Style (soweit möglich).

4.3.2.2.4.3.39.25.5. Style-Sheet-Eigenschaften (Style-Sheet-Formatangaben)

4.3.2.2.4.3.39.25.5.1. Style-Sheet und Wertzuweisung an Eigenschaften

eigenschaft: wert; oder eigenschaft=wert;

Hinweis: Pflichtkodierung von Doppelpunkt bzw. Gleichheitszeichen
Semikolon

Folge von Eigenschaften möglich: als Liste mit Semikolontrennung
Bsp: eigenschaft1=wert1; .....; eigenschaft_n=wert_n;

wenn Blank in Kodierung enthalten, so alles in " " setzen

Bsp: style="font-style**:** italic; color**:**red;"

4.3.2.2.4.3.39.25.5.2. Style-Sheet-Eigenschaften für HTML-Elemente

4.3.2.2.4.3.39.25.5.2.1. Style-Sheet und Auswertung von Pixelpositions-Angaben

position: absolute; Positionsangaben als bezüglich Anzeigefenster-Rand
Element ist scrollbar
position: fixed; Positionsangaben bezüglich Anzeigefenster-Rand
Element ist nicht scrollbar
position: relative; Positionsangaben als bezüglich Vorgänger-Element
position: static; hebt absolute bzw. fixed bzw. relative auf

4.3.2.2.4.3.39.25.5.2.2. Style-Sheet und Abstand von HTML-Elementen

top:abstand_oberhalb_in_pixel; oder top:auto;
left:abstand_links_in_pixel; oder left:auto;
bottom:abstand_unterhalb_in_pixel; oder bottom:auto;
right:abstand_rechts_in_pixel; oder right:auto;

padding-top: abstand_element_grenze_oben_zum_element_oberhalb_in_pixel;
padding-bottom: abstand_element_grenze_unten_zum_element_unterhalb_in_pixel;
padding-left: abstand_element_grenze_links_zum_element_links_in_pixel;
padding-right: abstand_element_grenze_rechts_zum_element_rechts_in_pixel;
padding: abstand; gilt für links, rechts, oben und unten zugleich

4.3.2.2.4.3.39.25.5.2.3. Style-Sheet und HTML-Element-Dimension (-Grenzen, -Anzeigebereich)

```
width: breite_in_pixel;                          oder    width:auto;
min-width: minimale_breite_in_pixel;    oder     min-width:auto;
max-width: maximale_breite_in_pixel;    oder     max-width:auto;

height: hoehe_in_pixel;                          oder    height:auto;
min-height: minimale_hoehe_in_pixel;    oder     min-height:auto;
max-height: maximale_hoehe_in_pixel;    oder     max-height:auto;

overflow:hidden;          wenn Elementgröße > max-width und max-heigt, so wird Element wird auf
                                Maximalwerte begrenzt
overflow:visible;         wenn Elementgröße > max-width und max-heigt , so werden Maximalwerte
                                ignoriert
```

4.3.2.2.4.3.39.25.5.2.4. Style-Sheet und HTML-Element-Sichtbarkeit

```
visibility:hidden;        unsichtbar, aber Platzhalter möglich
visibility:visible;       sichtbar

display:    none;         unsichtbar UND kein Platzhalter möglich
                          block;          sichtbar
                          inline;         sichtbar
```

4.3.2.2.4.3.39.25.5.2.5. Style-Sheet und Element-Ausschnitt

```
clip: rect (oben rechts unten links);   Pixelangaben bezüglich Elementgrenze
clip: auto;
```

4.3.2.2.4.3.39.25.5.2.6. Style-Sheet und Element-Scrolling

```
Scrolling nötig, wenn Elementgröße > max-width und max-heigt
                          ABER Maximalwerte eingehalten werden sollen

overflow:scroll;          Scrolling immer möglich
overflow:auto;

siehe auch overflow: visible;  bzw.  overflow hidden;
```

4.3.2.2.4.3.39.25.5.2.7. Style-Sheet und Layer-Element

```
z-index:index_nr;   Nummer innerhalb der Anzeigereihenfolge sich überlappender Elemente
                          ab 1 = unterste Schicht
```

4.3.2.2.4.3.39.25.5.2.8. Style-Sheet und HTML-Elemente-Anordnung im Dokument

```
direction:ltr;            von links nach rechts;  ist Standard
direction: rtl;           von rechts nach links

display:block;            Element in eigenem Absatz anzeigen
display:inline;           Element nicht in eigenem Absatz anzeigen
dsiplay:none;             Element nicht anzeigen
display:list-item;

float:left;         Element linksbündig; Textfluss rechts
float:right;        Element rechtsbündig; Textfluss links
float:none:         Standard

clear:left;         hebt float: left; auf
clear:right;        hebt float: right; auf
clear:both;         hebt float: left;  UND  float: right;  auf
clear:none;               identisch mit float: none;
```

4.3.2.2.4.3.39.25.5.2.9. Style-Sheet und Elemente-Anzeige als Aufzählungsliste (Bullet)

```
display: list-item;   Element in eigenem Absatz UND mit Bullet anzeigen
```

4.3.2.2.4.3.39.25.5.3. Style-Sheet-Eigenschaften für Liste (numerisch, Aufzählung)

```
list-style-type:    decimal;        1     2     3 ...
        oder        lower-roman;    i     ii    iii ..
        oder        lower-alpha     a     b     c ....
        oder        upper-roman;    I     II    III ...
        oder        upper-alpha;    A     B     C ....
        oder        disk;           Bullet ist Dateisymbol
        oder        circle;         Bullet ist rund
        oder        square;         Bullet ist rechteckig
        oder        none;

list-style-position:inside;         Listenelement einrücken; ist Standard
list-style-position:outside         Listenelement ausrücken

list-style-image: url(bullet_grafik_datei);   GIF oder JPG

list-style: liste_aller_obigen_eigenschaften_mit_blank_trennung;
```

4.3.2.2.4.3.39.25.5.4. Style-Sheet-Eigenschaften für Tabelle

caption-side:	top;	Überschrift oberhalb zentriert
oder	topleft;	Überschrift oberhalb linksbündig
oder	topright;	Überschrift oberhalb rechtsbündig
oder	bottom;	Überschrift unterhalb zentriert
oder	bottomleft;	Überschrift unterhalb linksbündig
oder	bottomright;	Überschrift unterhalb rechtsbündig

row-span: anzahl_der_zeilen_über_die_sich_die_zelle_ausstrecken_soll;
column-span: anzahl_der_spalten_über_die_sich_die_zelle_ausstrecken_soll;

4.3.2.2.4.3.39.25.5.5. Style-Sheet-Eigenschaften für Seitendarstellung im Dokument

margin-top: rand_breite_oben;	z.B. 2cm
margin-bottom: rand_breite_unten;	z.B. 2cm
margin-left: rand_breite_links;	z.B. 2cm
margin-right: rand_breite_rechts;	z.B. 2cm

margin: werte_liste_obiger_breiten_mit_blank_trennung;

wenn nur 1 Wert kodiert:	gilt für oben UND unten UND links UND rechts
wenn 2 Werte kodiert:	1. Wert gilt für oben UND unten
	2. Wert gilt für links UND rechts
wenn 3 Werte kodiert:	1. Wert gilt für oben
	2. Wert gilt für links UND rechts
	3. Wert gilt für unten
wenn 4 Werte kodiert:	1. Wert gilt für oben
	2. Wert gilt für rechts
	3. Wert gilt für unten
	4. Wert gilt für links

size:	seiten_breite;		z.B. 20cm
oder	seiten_hoehe;		z.B. 29cm
oder	seiten_breite seiten_hoehe;		Blanktrennung
oder	landscape;		Querformat
oder	portrait;		Hochformat
oder	auto;		
page-break_before:	always;		immer Seitenumbruch vor aktuellem Element erzeugen
	oder	aroid;	nie Seitenumbruch vor aktuellem Element erzeugen
	oder	left;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann linksbündig ablegen
	oder	right;	immer Seitenumbruch vor aktuellem Element erzeugen UND Element dann rechtsbündig ablegen
	oder	auto;	
page-break_after:	always;		immer Seitenumbruch nach aktuellem Element erzeugen
	oder	aroid;	nie Seitenumbruch nach aktuellem Element erzeugen
	oder	left;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element linksbündig ablegen
	oder	right;	immer Seitenumbruch nach aktuellem Element erzeugen UND Element rechtsbündig ablegen
	oder	auto;	

4.3.2.2.4.3.39.25.5.6. Style-Sheet-Eigenschaften für Text

text-ident: wert; wert > 0, so Textzeile einrücken
wert < 0, so Textzeile ausrücken

text-align:	left;	Text linksbündig
oder	center;	Text zentriert
oder	right;	Text rechtsbündig
oder	justify;	Text im Blocksatz
vertical-align:	top;	Text auf Oberkante der Umgebung
oder	middle;	Text auf Mitte der Umgebung
oder	bottom;	Text auf Unterkante der Umgebung
oder	sub;	Text tieferstellen
oder	super;	Text höher stellen
oder	baseline;	Text auf Basislinie des umgebenden Textes mit verschiedenen Schriftarten
oder	text-top;	Text auf obere Linie des umgebenden Textes mit verschiedenen Schriftarten
oder	text-bottom;	Text auf untere Linie des umgebenden Textes mit verschiedenen Schriftarten

font-size: schriftgroesse_wert; z.B. 2pt oder vordefiniert

line-height: zeilenhoehe_wert; z.B. 2pt* oder vordefiniert/

white-space: normal; automatischer Zeilenumbruch einschalten
oder pre; manuellen Zeilenumbruch einschalten
oder nowrap; kein Zeilenumbruch möglich

color:#rrggbb; oder rgb(rrr,ggg,bbb); auch vordefiniert möglich

colums: anzahl_der_text_spalten_fuer_textfluss;
column-gap: abstand_der_text_spalten_fuer_textfluss;
column-rule-width: dicke_des_trennstriches_zwischen_den_textspalten_vom_textfluss;
column-rule-color: #rrggbb; oder rgb(rrr,ggg,bbb); Farbe Trennstrich
column-rule-style: none; kein Trennstrich zwischen Textspalten
oder dotted; gepunkteter Trennstrich
oder dashed; gestrichelter Trennstrich
oder solid; einfacher durchgezogener Trennstrich
oder double; doppelter durchgezogener Trennstrich
oder groove; 3D-Effekt 1
oder ridge; 3D-Effekt 2
oder inset; 3D-Effekt 3
oder outset; 3D-Effekt 4
column-rule: werte_liste_aus_obigen_textfluss_werten_mit_blank_trennung;
word-spacing: abstand_zwischen_woertern_im_text;
letter-spacing: abstand_zwischen_zeichen_im_text;
text-decoration: underline; Text unterstrichen
oder overline; Text überstrichen
oder line-through; Text durchgestrichen
oder blink; Text blinkend
oder none; Text normal
text-transform: capitalize; Wortanfang ALLER Wörte auf Großbuchstaben
oder uppercase; alle Zeichen nach Großbuchstaben
oder lowercase; alle Zeichen nach Kleinbuchstaben
oder none;
text-shadow: #rrggbb; oder rgb(rrr,ggg,bbb); auch vordefiniert möglich
oder none; kein Textschatten

orphans: anzahl_der_VOR_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
Schusterjunge
widow: anzahl_der_NACH_seitenumbruch_zusammenzuhaltender_zeilen; Standard ist 2
Hurenkind

4.3.2.2.4.3.39.25.5.7. Style-Sheet-Eigenschaften für Font

font-family: schriftarten_liste_mit_kommatrennung;

es wird der ERSTE auf dem lokalen Rechner gefundene Font aus der Liste geladen
wenn blank vorhanden, so alles in " " setzen
vordefinierte Schriftarten: serif
san-serif
cursive
fantasy
monospace

font-style: italic; kursiv
oder oblique; kursiv
oder normal; nicht kursiv

font-variant: small-caps; kleine Großbuchstaben (Kapitälchen)
oder normal; kein Kapitälchen

font-size: schrift_groesse_wert;
oder xx-small; winzig
oder x-small; sehr klein
oder small; klein
oder medium; mittel
oder large; groß
oder x-large; sehr-groß
oder xx-large; riesig
oder smaller; etwas kleiner als normal
oder larger; etwas größer als normal

font-weight: bold; fett
oder bolder; extra fett
oder lighter; dünn
oder normal; nicht dünn und nicht fett
oder 100; extra dünn

oder	200;	
oder	300;	
oder	400;	
oder	500;	medium
oder	600;	
oder	700;	bold
oder	800;	
oder	900;	extra fett

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;

4.3.2.2.4.3.39.25.5.8. Style-Sheet-Eigenschaften für Mauscursor

cursor:	auto;	
oder	default;	
oder	crosshair;	Fadenkreuz
oder	pointer;	Zeiger
oder	move;	Kreuz für Beweglichkeit
oder	n-resize;	Pfeil Nord
oder	ne-resize;	Pfeil Nord-Ost
oder	e-resize;	Pfeil Ost
oder	se-resize;	Pfeil Süd-Ost
oder	s-resize;	Pfeil Süd
oder	sw-resize;	Pfeil Süd-West
oder	w-resize;	Pfeil West
oder	nw-resize;	Pfeil Nord-West
oder	text;	|
oder	wait;	Sanduhr
oder	help;	Fragezeichen
oder	url(url_oder_mauscursor_grafik_datei);	GIF oder JPG

4.3.2.2.4.3.39.25.5.9. Style-Sheet-Eigenschaften für Unicode (Zeichensatz)

unicode-range:U+xxxx-yyyy;

Fragezeichen als Joker innerhalb von xxxx und yyyy verwendbar

Bsp:	unicode-range:U+0000-007F;	ist ASCII-Zeichensatz
	unicode-range:U+0000-00?F;	? steht für 0 bis F --> ist nicht ASCII-Zeichensatz

4.3.2.2.4.3.39.25.6. Style-Sheet-Beispiele

Beispiel 1

```
<STYLE TYPE="text/css">
<!--
        h1{font-size:24pt;margin-top:1.2cm;margin-left:30px;}
        body {background-color:rgb(51,0,102);}
        p,li {font-size:12pt;line-height:14pt;font-family:Helvetica,Arail;}
        p.normal {fonszie:10pt;color:bllack;}
        p.klein {fontsize:8pt;color:black}
        all.rot {color.red;}    oder .rot {color:red}
                                        Rot-Definition für ALLE Tags
                                                Anwendung  z.B. per <P CLASS="normal">text</P>
                                                Anwendung  z.B. per <P CLASS="rot">text</P>
                                                Anwendung  z.B. per <H1 CLASS="rot">text</H1>
        #fett_kursiv {font-weight:bold;font-style:italic;}
                                        Anwendung z.B. per <P ID="fett_kursiv">text</P>
        a:link {color:#FF0000;font-weight:bold;}
                                        Anwendung z.B. per <A HREF="....." >text</A>
// -->
</STYLE>
```

Anwendung je nach Tag --> meist innerhalb <BODY>

Beispiel 2

```
<BODY>
        ...
        <DIV STYLE="background-color:#FF0000;"> ..... </DIV>
        <P>
                text1
                <SPAN   STYLE="color:red;">
                        text2_in_rot;
                </SPAN>
                text3_wie_text1
        </P>
....
```


</BODY>

4.3.2.2.4.3.39.26. Beispiele für Objekteigenschaft .style

Kodierung der Style-Sheets:
innerhalb CSS-Klasse: immer mit Bindestrich
als .style-Eigenschaft ohne Bindestrich aber anstelle dessen die Nachfolgebuchstaben als Großbuchstabe

Bsp: in CSS-Klasse background-color
.style.backgroundColor

alle Eigenschaften sind les- und schreibbar

Die Erweiterung von .style durch HTML-CSS-konforme Style-Sheets ist möglich, aber zu prüfen, ob sie auch unterstützt werden. Beachte die o.g. Kodierungsregel.

4.3.2.2.4.3.39.26.1. Zeichensatz- und Texteigenschaften

font: liste_aller_obiger_eigenschaften_mit_blank_trennung;
Liste darf maximal 6 der nachfolgenden Eigenschaften beinhalten

caption	Zeichnsatz für Objekte mit Überschriften
font-style	
font-variant	
font-weight	
font-size	
line-height	
font-family	
icon	Zeichensatz für Grafik
menu	Zeichensatz in Menüs
message-box	Zeichensatz in Mesages-Boxen
small-caption	Zeichensatz für Control-Elemente
status-bar	Zeichensatz für Stautszeile

font-family: schriftarten_liste_mit_kommatrennung;
es wird der ERSTE auf dem lokalen Rechner gefundene Font aus der Liste geladen
wenn blank vorhanden, so alles in " " setzen
vordefinierte Schriftarten: serif
san-serif
cursive
fantasy
monospace

font-size:	xx-small		winzig
	oder	x-small	sehr klein
	oder	small	klein
	oder	medium	mittel
	oder	large	groß
	oder	x-large	sehr-groß
	oder	xx-large	riesig
	oder	smaller	etwas kleiner als normal
	oder	larger	etwas größer als normal
	oder	Fliesskommazahl mit Einheit z.B. mm oder cm oder pt	
	oder	Prozent z.B. 3%	
font-style:	normal		nicht kursiv
	oder	vordefinierte	z.B. italic; kursiv
font-variant:		small-caps	kleine Großbuchstaben (Kapitälchen)
	oder	normal	kein Kapitälchen
font-weight:		bold	fett
	oder	bolder	extra fett
	oder	lighter	dünn
	oder	normal	nicht dünn und nicht fett
	oder	100	extra dünn
	oder	200	
	oder	300	
	oder	400	
	oder	500	medium
	oder	600	
	oder	700	bold
	oder	800	
	oder	900	extra fett

letter-spacing:		Abstand zwischen Zeichen im Text	
		normal	
	oder	Fliesskommazahl mit Einheit z.B. mm oder cm oder pt	
line-height:		Abstand zwischen 2 Zeilen	
		normal	
	oder	Fliesskommazahl mit Einheit z.B. mm oder cm oder pt	
text-align:	left	Text linksbündig	
	oder	center	Text zentriert
	oder	right	Text rechtsbündig
	oder	justify	Text im Blocksatz
text-decoration:		underline	Text unterstrichen
	oder	overline	Text überstrichen
	oder	line-through	Text durchgestrichen
	oder	blink	Text blinkend
	oder	none	Text normal
text-indent:		Texteinschub	
		Fliesskommazahl mit Einheit z.B. mm oder cm oder pt	
	oder	Prozent z.B. 3%	
text-transform:		capitalize	Wortanfang ALLER Wörte auf Großbuchstabe
	oder	uppercase	alle Zeichen nach Großbuchstabe
	oder	lowercase	alle Zeichen nach Kleinbuchstabe
	oder	none	
white-space:		Zeilenumbruch ab IE 5.x	
		normal	automatischer Zeilenumbruch einschalten
	oder	pre	manuellen Zeilenumbruch einschalten
	oder	nowrap	kein Zeilenumbruch möglich
word-wrap:		Wortumbruch ab IE 5.x	
		normal	
	oder	break-word	

4.3.2.2.4.3.39.26.2. Farben- und Hintergrundeigenschaften

background:		Liste folgender Eigenschaften in fester Reihenfolge und Blanktrennung	
		color	wie background-color
		image	wie background-image
		repeat	wie background-repeat
		attachment	wie background-attachment
		position	wie background-position
	oder	"null"	transparenter Hintergrund
background-attachment:		scroll	Hintergrundbild scrollt mit dem Objekt z.B. dem Dokument
	oder	fixed	Hintergrundbild ist fixiert
background-color:		Hintergrundfarbe	
		#rrggbb	
	oder	rgb(rrr,ggg,bbb)	
	oder	vordefiniert	
background-image:		none	kein Hintergrundbild
	oder	url("pfad_und_grafikdatei_name_als_zeichenkette")	
background-repeat:		repeat	Hintergrundbild horizontal und vertikal wiederholen
	oder	no-repeat	Hintergrundbild nicht horizontal und nicht vertikal wiederholen
	oder	repeat-x	Hintergrundbild horizontal wiederholen
	oder	repeat-y	Hintergrundbild vertikal wiederholen
color:		Textfarbe (Vordergrundfarbe)	
		#rrggbb	
	oder	rgb(rrr,ggg,bbb)	
	oder	vordefiniert	

4.3.2.2.4.3.39.26.3. Layouteigenschaften

border:	Rahmen ziehen auf allen 4 Seiten	
	Eigenschaftenliste mit fester Reihenfolge mit Blanktrennung	
	width	wie border-width
	style	wie border-style
	color	wie border-color
border-top:	wie border aber für oberen Rahmenteil	
border-bottom:	wie border aber für unteren Rahmenteil	

border-left:	wie border aber für linken Rahmenteil	
border-right:	wie border aber für rechten Rahmenteil	
border-color:	Rahmenfarbe aller Rahmenteile	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)
	oder	vordefiniert
border-top-color:	wie border-color aber für oberen Rahmenteil	
border-bottom-color:	wie border-color aber für unteren Rahmenteil	
border-left-color:	wie border-color aber für linken Rahmenteil	
border-right-color:	wie border-color aber für rechten Rahmenteil	
border-style:	Stil aller Rahmenteile	
	none	kein rahmen
	oder	dotted gepunktet
	oder	dashed gestrichelt
	oder	solid solid
	oder	double doppelter
	oder	groove 3D-Rahmen 1
	oder	ridge 3D-Rahmen 2
	oder	inset 3D-Rahmen 3
	oder	outset 3D-Rahmen 4
border-bottom-style:	wie border-style aber für unteren Rahmenteil	
border-left-style:	wie border-style aber für linken Rahmenteil	
border-right-style:	wie border-style aber für rechten Rahmenteil	
border-width:	Dicke aller Rahmenteile	
	medium	
	oder	thin
	oder	thick
	oder	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
border-top-width:	wie border-width aber für oberen Rahmenteil	
border-bottom-width:	wie border-width aber für unteren Rahmenteil	
border-left-width:	wie border-width aber für linken Rahmenteil	
border-right-width:	wie border-width aber für rechten Rahmenteil	
margin:	Randabstand auf allen 4 Seiten	
	auto	
	oder	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
	oder	Prozentangabe
margin-top:	wie margin aber für oberen Abstand	
margin-bottom:	wie margin aber für unteren Abstand	
margin-left:	wie margin aber für linken Abstand	
margin-right:	wie margin aber für rechten Abstand	
padding:	Abstand Objektinhalt zum Objektrand auf allen 4 Seiten	
	auto	
	oder	Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
	oder	Prozentangabe
padding-top:	wie padding aber für oberen Abstand	
padding-bottom:	wie padding aber für unteren Abstand	
padding-left:	wie padding aber für linken Abstand	
padding-right:	wie padding aber für rechten Abstand	
scrollbar3d-light-color:	Scrollbar-Farbe bei 3D	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)
	oder	vordefiniert
scrollbar-arrow-color:	Scrollbar-Pfeile-Farbe ab IE 5.x	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)
	oder	vordefiniert
scrollbar-base-color:	Scrollbar-Basis-Farbe ab IE 5.x	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)
	oder	vordefiniert
scrollbar-dark-shadow-color:		
	Scrollbar-Farbe des dunklen Schatten ab IE 5.x	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)
	oder	vordefiniert
scrollbar-face-color:	Scrollbar-Face-Farbe ab IE 5.x	
	#rrggbb	
	oder	rgb(rrr,ggg,bbb)

oder vordefiniert

scrollbar-highlight-color: Scrollbar-Highlight-Farbe ab IE 5.x
#rrggbb
oder rgb(rrr,ggg,bbb)
oder vordefiniert

scrollbar-shadow-color: Scrollbar-Farbe des Schatten ab IE 5.x
#rrggbb
oder rgb(rrr,ggg,bbb)
oder vordefiniert

zoom: Objekt vergrößern ab IE 5.x
normal
oder Fliesskommazahl als Faktor (1,0 ist normal)
oder Prozentangabe mit % z.B. 50% (100% ist normal)

4.3.2.2.4.3.39.26.4. Positionierungsangaben

bottom: Abstand zum oben umgebenden Objekt ab IE 5.x
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe

display Sichtbarkeit
none unsichtbar
block anzeigen
inline anzeigen

left: Abstand zum links umgebenden Objekt ab IE 5.x
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe
beachte Wert der Eigenschaft document.all.position

top: Abstand zum unten umgebenden Objekt
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe
beachte Wert der Eigenschaft document.all.position

right: Abstand zum rechts umgebenden Objekt ab IE 5.x
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe

offset-left: wie left
offse-top: wie top

pixel-left: absolute X-Position der linken oberen Objekt-Ecke im Pixelsystem,
dessen Ursprung (0,0) links oben liegt
also Abstand vom linken Fensterrand
nur Integerzahl ohne Einheit, da automatisch in Pixel interpetiert

pixel-top: absolute Y-Position der linken oberen Objekt-Ecke im Pixelsystem,
dessen Ursprung (0,0) links oben liegt
also Abstand vom oberen Fensterrand
nur Integerzahl ohne Einheit, da automatisch in Pixel interpetiert

height: Höhe des Objektes
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe
Hinweis: wenn geändert wird, so automatischer Umbruch des
Inhaltes des HTML-Elementes

width: Breite des Objektes
auto
oder Fliesskommazahl mit Einheitenangabe z.B. pt oder cm
oder Prozentangabe

Hinweis: wenn geändert wird, so automatischer Umbruch des
Inhaltes des HTML-Elementes

visibility : Objektsichtbarkeit
visible sichtbar
oder hidden unsichtbar
oder inherit Sichtbarkeit laut Elternobjekt

z-index: auto jede neu erzeugte Objekt ist das oberstes

Achtung: sollte die Dimension des neuen Objektes
sich nach der Erzeugung derart verändern,
so dass es ein anderes Objekt überlagert,
dann ändert sich trotzdem nichts an der
Sichtbarkeit: da neue Objekt liegt unter
dem vorhandenen Objekt, das vom neuen
Objekt nun überlagert wird, denn zum
Zeitpunkt der Erzeugung des neuen Objektes
war keine Überlagerung ---> anstelle auto
immer z-index kodieren !!!!

oder Schichtnummer des Objektes bei sich irgendwann überlagernden Objekten
(also auch bei Überlagerung, die erst nach deren Erzeugung
erfolgt)
ganzzahlig, also auch < 0
je kleiner um so tiefer in der Schicht
je höher um so höher in der Schicht
Sichtbar ist immer die oberste, also höchste Schicht
wenn 2 Objekte mit identischer Schichtnummer, so
Sichtbarkeit laut Reihenfolge der Kodierung
im Quelltext

4.3.2.2.4.3.39.26.5. Druckeigenschaften

page-break-after:	wenn Druck des Objektes vollzogen, so soll Seitenumbruch erfolgen		
		always	ja
	oder	auto	Browser entscheidet
	oder	""	nein
page-break-before:	bevor der Druck des Objektes beginnt, so soll Seitenumbruch erfolgen		
		always	ja
	oder	auto	Browser entscheidet
	oder	""	nein

4.3.2.2.4.3.39.26.6. Cursor

cursor:		auto	
	oder	crosshair (Kreuz)	
	oder	default	je nach Windowseinstellung
	oder	hand	
	oder	move	Verschieben-Symbol in alle Richtungen
	oder	N-resize	Verschiebe-Symbol in Richtung oben
	oder	NE-resize	Verschiebe-Symbol in Richtung oben-rechts
	oder	NW-resize	Verschiebe-Symbol in Richtung oben-links
	oder	S-resize	Verschiebe-Symbol in Richtung unten
	oder	SE-resize	Verschiebe-Symbol in Richtung unten-rechts
	oder	SW-resize	Verschiebe-Symbol in Richtung unten-links
	oder	E-resize	Verschiebe-Symbol in Richtung rechts
	oder	W-resize	Verschiebe-Symbol in Richtung links
	oder	text	Strich
	oder	wait	Sanduhr
	oder	help	Fragezeichen

Achtung: unter Windows sind Cursor-Belegungen veränderbar !!!
Bsp: anstelle Sanduhr ein anderes Symbol, dass dann natürlich
auch bei wait erscheint

4.3.2.2.4.3.39.26.7. Bildausschnitt

clip enthält die Form UND Dimension des Bildausschnittes
auf der Ebene des HTML-Elementes
wird eigentlich nur zum Schreiben verwendet
Form des Bildaussschnittes: RECHTECKIG
Bsp.zu schreiben:
.clip='rect:(x1,y1,x2,y2)';
x1,y1 linke obere Ecke
x1 horizontal in Pixel
y1 vertikal in Pixel
x2,y2 rechte untere Ecke
x2 horizontal in Pixel
y2 vertikal in Pixel
x1 bis y2 immer bezüglich links oben im
HTML-Element
bzw.im Fenster des Browsers
können den Wert auto haben, der immer die
maximale Dimension bewirkt
Bsp. 'rect:(auto,y1,auto,y2)';
wenn alle auf auto, so wird der
Bildausschnitt maximiert,
was der Standardimension

entspricht

4.3.2.2.4.3.40. styleSheet Objekt des Internet Explorer

ab IE 4.x

Ansatz:

Dieses Objekt dient der Verwaltung von Styles (StyleSheets) im Dokument, die auf ganz bestimmte Arten in das Dokument implementiert wurden:

Beispiele für Arten:

```
<HEAD>
<STYLE>
        @import url("MeineStyleSheetDatei.css");
</STYLE>
<HEAD>

<LINK REL=stylesheet HREF="styles.css" type="text/css">

per Pseudoklasse @page (Objekt page)

<HEAD>
<STYLE>
        P {color:green}
</STYLE>
</HEAD>

per document.createStyleSheet('styles.css');
```

Diese o g. Arten von Implementationen können als Block von Styles angesehen werden, mit anderen Worten: Als Menge von Style-Regeln.

Besonders für den Programmierer ist die vereinfachte Kodierung von Styles in externen CSS-Dateien und die Verwendung der browsereigenenen Pseudoklassen wie @page und @import interessant: Anhand dieser sind komplette Style-Regeln-Manipulationen zur Laufzeit des HTML-Dokumentes möglich. So kann z.B. das Layout des ganzen HTML-Dokumentes nach seinem kompletten Laden auf einen Schlag geändert werden.

Allen Arten von Style-Implementationen haben eines gemeinsam: Es werden einzelne Styles im Browser verwaltet, die auch alle einzeln durch die übliche Style-Eigenschaften und Style-Methoden verwaltet werden könnten (siehe style Objekt).

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem
Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !!!

Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:
Die zuletzt während der Laufzeit getätigte Defintion ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

Verwaltung des styleSheet-Objektes in der Collection document.styleSheets:

Ein styleSheet-Objekt ist ein Element der Collection document.styleSheets, die in der Reihenfolge der Style-Implemenationen im Quellcode der HTML-Seite beim Laden des Dokumentes gefüllt, also initialsiert wird. Jede o.g. Implementation erzeugt je ein styleSheet-Objekt als Element der Collection dodument.styleSheets.

Beispiel:

```
<HEAD>
<STYLE>
        .StyleRegel1 {color:"red"}
        .StyleRegel2 {color:"blue"}
</STYLE>
<HEADY
```

Es wird ein styleSheet-Objekt erzeugt ,das 2 Regeln besitzt.

Verwaltung der Implementationsarten im styleSheet-Objekt:

Im styleSheet-Objekt wird jede o.g. Art der Style-Implementation einzeln verwaltet und zwar mit je einer eigenen Collection des styleSheet-Objektes.Ein styleSheet-Objekt referenziert selbst diese Collectionen, die je eine vordefinierte Art der Style-Implementation in das Dokument verwalten. Ob diese Collectionen auch reale Elemente besitzen, hängt davon ab, ob Styles in o.g. Arten implementiert wurden oder noch werden.

Collectionen des styleSheet Objektes Beispiele zu den Arten der verwalteteten Style-Implementationen

styleSheet.imports	<HEAD> <STYLE> @import url("MeineStyleSheetDatei.css"); </STYLE> <HEAD> <LINK REL=stylesheet HREF="styles.css" type="text/css"> document.createStyleSheet('styles.css'); Methode .addImport()
styleSheet.pages	per Pseudoklasse @page (Objekt page)
styleSheet.rules	<HEAD> <STYLE> P {color:green} </STYLE> </HEAD>

Damit weist ein Eintrag in der Collection document.styleSheets auf ein styleSheet Objekt, das wiederum genau die zur Art der Implementation passende Collection referenziert, die wiederum konkrete Style-Objekte referenziert, welche eben nur auf eine von o.g. Arten der Implementationen in das Dokument eingebettet wurden und sich ansonsten **nicht** z.B. von einem inline-Style (per STYLE-Attribut im HTML-Tag eingebetter Style) unterscheiden.

Beispiel:

```
<HEAD>
<STYLE>
        .StyleRegel1 {color:"red"}
        .StyleRegel2 {color:"blue"}
</STYLE>
<HEADY
```

Implementation füllt Collection styleSheet.rules mit 2 Elementen (2 Regeln)

Wenn obige Implementation die ERSTE im Dokument ist, so wird das ERSTE styleSheet-Objekt erzeugt, also das Element document.styleSheets[0], das auf die Collection styleSheet.rules verweist.

Zugriff auf styleSheet-Objekt:
nur per Collection document.styleSheets

Hinweis: Style, der per Link oder @import anhand einer CSS-Datei eingebunden, kann zur Laufzeit **nur** verändert werden, wenn document.designMode auf "On" gesetzt ist

Beispiel 1:

```
<HEAD>
<STYLE>
        BODY {background-color: #CFCFCF;}
        @import url("MeineStyleSheetDatei.css");
</STYLE>
<SCRIPT>
        window.onload=fnInit;

        function fnInit()
        {
                // Style des BODY laut Klassendefinition
                var StyleSheetObjekt =document.styleSheets[0];

                // erste Regel zum Style des BODY holen
                var RegelObjekt = StyleSheetObjekt.rules[0];

                // und Style ändern
                RegelObjekt.style.backgroundColor="#0000FF";

                // neue Regel für BODY hinzufügen
                StyleSheetObjekt.addRule("BODY"," border-color: #FFFF00;");

                // und weitere neue Regel für BODY
                StyleSheetObjekt.imports[0].color="#000000";
        }
</SCRIPT>
</HEAD>
```

Beispiel 2:

```
<HTML>
<HEAD>
<SCRIPT>
	function Anzeige(RegelIndex)
	{
		alert(		"Regel Nr " + RegelIndex
				+ " hat style.color =  "
				+ document.styleSheets[0].rules.item(RegelIndex).style.color
				+ "."
			);
	}
</SCRIPT>
<STYLE>
	.StyleRegel1 {color:"red"}
	.StyleRegel2 {color:"blue"}
</STYLE>
</HEAD>
<BODY>
	<P CLASS=" StyleRegel1">
		StyleRegel1
	</P>
	<P CLASS=" StyleRegel2">
		StyleRegel2
	</P>

	<BUTTON onclick="Anzeige(0)">StyleRegel1</BUTTON>
	<BUTTON onclick="Anzeige(1)"> StyleRegel2</BUTTON>
</BODY>
</HTML>
```

Zugriff auf Collectionen des styleSheet-Objektes:

siehe Beschreibung der Collectionen

Eigenschaften des styleSheet-Objektes:

.href	Url einer externen Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
.owningElement	Referenz auf den im StyleSheet implementierten Style als Kindobjekt Hinweis: Es sind die Eigenschaften und Methoden des style Objektes referenzierbar

Beispiel:

```
for ( var i = 0; i < document.styleSheets.length; i++ )
{
	if ( document.styleSheets(i).owningElement.tagName == "STYLE" )
	{
		for ( var j = 0; j < document.styleSheets(i).imports.length; j++ )
		{
			alert(		"Importierter StyleSheet (Style-Regel) Nr " + j
					+ " hat HREF = " + document.styleSheets(i).imports(j).href
				);
		}
	}
}
```

.parentStyleSheet	Referenz auf das den StyleSheet implementierende Objekt (Elternobjekt)
.readOnly	Art der Implementation des StyleSheets im Dokument ermitteln
.title	Titel des StyleSheets
.type	Mimetyp des StyleSheets muss "text/css" sein

Methoden des styleSheet-Objektes:

.addImport()	StyleSheet aus externer CSS-Datei importieren und als Element der Collection styleSheet.imports erzeugen (entspricht @import url() innerhalb STYLE im HEAD)
.addPageRule()	StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.pages erzeugen (entspricht @page vom Objekt page)

Beispiel:

```
function TextFaerben ()
{document.styleSheets[0].addPageRule("DIV B", "color:blue", 0);}

<DIV onmouseover="TextFaerben ();">
	<B>dieser Text wird per CSS mit blauer Fabrbe angezeigt</B>
</DIV>
```

.addRule()	StyleSheet importieren, als ob er wie ein in das Dokument direkt kodierter Style implementiert wurde, und als Element der Collection styleSheet.rules erzeugen (entspricht xx { ...} innerhalb STYLE im HEAD)

Beispiel:

```
function TextFaerben ()
{document.styleSheets[0].addRule("DIV B", "color:blue", 0);}
```


```
<DIV onmouseover="TextFaerben ();">
    <B>dieser Text wird per CSS mit blauer Fabrbe angezeigt</B>
</DIV>
```

.fireEvent() ein Event auslösen
true Event erfolgreich ausgelöst
false Event nicht ausgelöst

.removeRule() StyleSheet aus der Collection styleSheet.rules entfernen
Achtung: Um Style auch sichtbar zu entfernen, muss
Dokument muss neu geladen werden
oder alle betroffene Style-Elemente neu mit Wert belegen
durch Zuweisung auf sich selbst (siehe Beispiel)

Beispiel:

```
<STYLE>
    P {color:green}
</STYLE>
 <SCRIPT>
    function StyleEntfernen()
    {
        var StyleSheetsObjekt = document.styleSheets;
        var AnzahlStyleSheets = StyleSheetsObjekt.length;

        if (AnzahlStyleSheets > 0)
        {

            // StyleSheet mit Index 0 bearbeiten also P {color:green}
            var StyleSheetAnIndex0 = StyleSheetsObjekt[0];

            //  wobei P {color:green} eine Regel ist, also Collection rules verwenden
            var StyleSheetsRegelCollection = StyleSheetAnIndex0.rules;

            var AnzahlRegeln = StyleSheetsRegelCollection.length;

            if (AnzahlRegeln > 0)
            {
                // Regel P {color:green} enfernen
                StyleSheetAnIndex0.removeRule(0);

                // visuell auch die Farbe entfernen durch Zuweisung auf sich selbst also
                //        nun ohne die Regel P {color:green}
                ID_P.innerHTML= ID_P.innerHTML;
            }
        }
    }
</SCRIPT>

<P ID="ID_P" >Test</P>
<BUTTON onclick="StyleEntfernen()">Text im P-Tag entfaerben.</BUTTON>
```

4.3.2.2.4.3.40.1. styleSheet.imports Collection des Internet Explorer

verwaltet die per

Link auf eine externe Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
@import einer externen Style-Sheet-Ressource-Datei (externe Style-Datei *.css)
Methode .addImport()

importierten Style-Regeln

z.B.
```
<HEAD>
<STYLE>
    @import url("MeineStyleSheetDatei.css");
</STYLE>
<HEAD>
```

z.B. <LINK REL=stylesheet HREF="styles.css" type="text/css">

z.B. document.createStyleSheet('styles.css');

Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Syntax:

[var FeldZeiger =] document.styleSheets**[**Index1**]**.imports
[var FeldElementZeiger =] document.styleSheets**[**Index1**]**.imports**[**Index2**]**

Index1: Integer und ab 0
muss in **[ ]** kodiert sein

Index2: Integer und ab 0
muss in **[ ]** kodiert sein

Beispiel:

```
for ( var i = 0; i < document.styleSheets.length; i++ )
{
    if ( document.styleSheets(i).owningElement.tagName == "STYLE" )
    {
        for ( var j = 0; j < document.styleSheets(i).imports.length; j++ )
        {
            alert(      "Importierter StyleSheet (Style-Regel) Nr " + j
                    + " hat HREF = " + document.styleSheets(i).imports(j).href
                );
        }
    }
}
```

Eigenschaften:

.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.3.40.2. page Objekt des Internet Explorer

repräsentiert eine @page Regel in einem styleSheet Objekt
ab IE 5.5

@page Regel:

Regel für Dimensionen, Orientierungen und Margins in einer Seite per styleSheet Objekt
Regel ist eine Pseudoklasse von Style
Seite wird als Rechteckbereich aufgefasst, der eingeteilt ist in:
Seitenbereich: Inhalt der Seite z.B. Text, Grafik
Marginbereich: Rand um den Seitenbereich

Syntax:

HTML @page Kette1 { Kette2 }

Kette1	":first"	Regel gehört der 1. Seite
	":footer"	Regel gehört der Fußnote
	":header"	Regel gehört der Kopfnote
	":left"	Regel gehört der linken Seite
	":left : header"	Regel gehört der Kopfnote Seite links
	":left : footer"	Regel gehört der Fußnote Seite links
	":right"	Regel gehört der rechten Seite
	":right:header"	Regel gehört der Kopfnote Seite rechts
	":right:footer"	Regel gehört der Fußnote Seite rechts
Kette2	String aus Regelfolge mit Semikolontrennung	

Beispiel: @page : left {font-weight:bold;font_style:italic;}

Eigenschaften:

.pseudoClass	Pseudoklasse einer Seite oder @page Regel per Objekt page
.selector	Seiten-Selektor per Objekt page

Methoden:

keine

4.3.2.2.4.3.40.3. styleSheet.pages Collection des Internet Explorer

verwaltet die per Pseudoklasse @page (Objekt page) importierten Style-Regeln

Syntax:

[var FeldZeiger =] document.styleSheets**[**Index1**]**.pages
[var FeldElementZeiger =] document.styleSheets**[**Index1**]**.pages**[**Index2**]**

Index1: Integer und ab 0
muss in **[ ]** kodiert sein

Index2: Integer und ab 0
muss in **[ ]** kodiert sein

Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Eigenschaften:

.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...>

4.3.2.2.4.3.40.4. styleSheet.rules Collection des Internet Explorer

verwalten die per

xxx { ...} innerhalb von STYLE im HEAD
oder per .addRule()

importierten Style-Regeln

z.B.
```
<HEAD>
<STYLE>
    P {color:green}
</STYLE>
</HEAD>
```
Hinweis: Für ein Element der Collection sind die Eigenschaften und Methoden des style Objektes referenzierbar

Syntax:

[var FeldZeiger =] document.styleSheets**[**Index1**]**.rules
[var FeldElementZeiger =] document.styleSheets**[**Index1**]**.rules**[**Index2**]**

Index1: Integer und ab 0
muss in **[ ]** kodiert sein

Index2: Integer und ab 0
muss in **[ ]** kodiert sein

Beispiel:
```
<HTML>
<HEAD>
<SCRIPT>
    function Anzeige(RegelIndex)
    {
        alert(      "Regel Nr " + RegelIndex
                + " hat style.color =  "
                + document.styleSheets[0].rules.item(RegelIndex).style.color
                + "."
            );
    }
</SCRIPT>
<STYLE>
    .StyleRegel1 {color:"red"}
    .StyleRegel2 {color:"blue"}
</STYLE>
</HEAD>
<BODY>
    <P CLASS=" StyleRegel1">
        StyleRegel1
    </P>
    <P CLASS=" StyleRegel2">
        StyleRegel2
    </P>

    <BUTTON onclick=" Anzeige(0)">StyleRegel1</BUTTON>
    <BUTTON onclick=" Anzeige(1)">StyleRegel2</BUTTON>
</BODY>
</HTML>
```
Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des

4.3.2.2.4.3.41. document.styleSheets Collection des Internet Explorer

Feld der Zeiger aller styleSheet Objekte im Dokument (siehe dort)

Syntax:

[var ZeigerAufFeld =] document.styleSheets
[var ZeigerAufFeldElement =] document.styleSheets **[**Index [, SubIndex]**]**

Index Integer ab 0
oder String Name oder ID des Elementes
muss in **[ ]** kodiert sein

SubIndex optional
nur kodieren wenn Index ein String ist
Integer als Unterindex also Unterelement eines Elementes

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !
Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:
Die zuletzt während der Laufzeit getätigte Defintion ersetzt wertmäßig den aktuellen Attributwert wenn gleiche Attributnamen/Eigenschaften betroffen sind (sonst hinzufügen).

Eigenschaften:

.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. <B> dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. <B> dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.length	Anzahl der Feldelemente also Feldlänge z.B. bei Collection
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.sourceIndex	Index des Objektes in der Collection document.all

Methoden:

.item()	Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!
.namedItem()	Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern
.urns()	Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

4.3.2.2.4.3.42. table Objekt des Internet Explorer

Objekt einer Tabelle (HTML-Element TABLE)

4.3.2.2.4.3.42.1. Erzeugung der Tabelle

4.3.2.2.4.3.42.1.1. Erzeugung in HTML

Die Tabelle kann folgende Tabellenelemente (Tags) besitzen:
CAPTION,
COL,
COLGROUP,
TBODY,
TD,
TFOOT,
TH,
THEAD
TR

wobei in der Tabelle maximal
1 THEAD
1 TFOOT
1 CAPTION (Überschrift)
auftauchen kann

TBODY-Tag nicht kodiert werden muss, wenn keine Fehlzuordung zu THEAD **und** kein TFOOT möglich ist

Beispiel:

```
<TABLE BORDER=1 WIDTH=80% BGCOLOR="gray">
<THEAD BGCOLOR="blue">
    <TR>
        <TH>Titel Spalte 1</TH>
        <TH>Titel Spalte 2</TH>
    </TR>
</THEAD>
<TBODY BGCOLOR="yellow">
    <TR>
        <TD>Zeile 1, Spalte 1 </TD>
        <TD>Zeile 1, Spalte 2 </TD>
    </TR>
    <TR>
        <TD>Zeile 2, Spalte 1</TD>
        <TD>zeile 2, Spalte 2</TD>
    </TR>
</TBODY>
```