

CGI mit C

Common Gateway Interface

- CGI Scripte dienen der dynamischen Erzeugung von Webseiten, der html-Code steht nicht in einer html-Datei, sondern wird von einem Programm dynamisch erzeugt. Der Aufruf dieses Programms erfolgt durch den http-Server, wenn er einen entsprechenden Link bekommt.
- CGI Scripte werden serverseitig ausgeführt, im Gegensatz zu Applets
- CGI Scripte können mit den verschiedensten Werkzeugen hergestellt werden, häufige Werkzeuge sind Pearl, C und Shellscriptsprachen

Apache2 konfigurieren

- Konfigurierung in /etc/apache2
- Konfigurationsdateien per link verknüpfen oder kopieren
 - mods-available → mods-enabled
 - userdir.conf, userdir.load
 - cgi.load, cgid.load, cgid.conf
- Einfügen in userdir.conf:
Options +ExecCGI
AddHandler cgi-script .cgi .pl

Verzeichnisse

- Dokumentenwurzel: /var/www oder /srv/www
- CGI-Verzeichnis: /usr/lib/cgi-bin oder unter der Dokumentenwurzel
- Dokumente im User-Dir: public_html
- CGI im User-Dir: public_html/cgi-bin
- Rechte setzen (755)

```
#include <stdio.h>
#include <ctype.h>
```

```
int main()
{
    int i,j;
    printf("Content-Type: text/html");
    printf("\n\n") ;
    puts("<html><head><title>CGI-Script - ASCII Codetabelle</title></head>\n");
    puts("<body><h2><hr>\n");
    puts("<pre>\n");

    for (i=0; i<4; i++)printf("|dec hex Char ");
    printf("\n");
    for (i=0; i<32; i++)
    {
        printf("\n| ");
        for (j=0; j < 128; j += 128/4)
        {
            printf("%3d %2X ",i+j,i+j);  if (isgraph(j+i))printf(" %c | ",j+i); else printf(" . | ");
        }
    }
    puts("</pre><hr>");
    puts("<a href=\"http://www.informatik.htw-dresden.de/~beck/a.beck.html\">A. Beck</A> <p>");
    puts("</body>\n");
    puts("</html>\n");
    return 0;
}
```

Arbeitsschritte:

- Quelltext erfassen
 - Compilieren gcc ascii.c -o ascii.cgi
 - nach cgi-bin kopieren
 - Lese-/Ausführungsrechte für alle für ascii.cgi
 - Aufruf über Browser
- <http://www.htw-dresden.de/~s12355/cgi-bin/ascii.cgi>

html-Quelltext wird
Über die Standardausgabe
An den Browser ausgegeben

<http://ipc145.informatik.htw-dresden.de/~beck/cgi-bin/ascii.cgi>

dec	hex	Char	dec	hex	Char	dec	hex	Char	dec	hex	Char
0	0	.	32	20	.	64	40	@	96	60	`
1	1	.	33	21	!	65	41	A	97	61	a
2	2	.	34	22	"	66	42	B	98	62	b
3	3	.	35	23	#	67	43	C	99	63	c
4	4	.	36	24	\$	68	44	D	100	64	d
5	5	.	37	25	%	69	45	E	101	65	e
6	6	.	38	26	&	70	46	F	102	66	f
7	7	.	39	27	'	71	47	G	103	67	g
8	8	.	40	28	(72	48	H	104	68	h
9	9	.	41	29)	73	49	I	105	69	i
10	A	.	42	2A	*	74	4A	J	106	6A	j
11	B	.	43	2B	+	75	4B	K	107	6B	k
12	C	.	44	2C	,	76	4C	L	108	6C	l
13	D	.	45	2D	-	77	4D	M	109	6D	m
14	E	.	46	2E	.	78	4E	N	110	6E	n
15	F	.	47	2F	/	79	4F	O	111	6F	o
16	10	.	48	30	0	80	50	P	112	70	p
17	11	.	49	31	1	81	51	Q	113	71	q
18	12	.	50	32	2	82	52	R	114	72	r
19	13	.	51	33	3	83	53	S	115	73	s
20	14	.	52	34	4	84	54	T	116	74	t
21	15	.	53	35	5	85	55	U	117	75	u
22	16	.	54	36	6	86	56	V	118	76	v
23	17	.	55	37	7	87	57	W	119	77	w
24	18	.	56	38	8	88	58	X	120	78	x
25	19	.	57	39	9	89	59	Y	121	79	y
26	1A	.	58	3A	:	90	5A	Z	122	7A	z
27	1B	.	59	3B	;	91	5B	[123	7B	{
28	1C	.	60	3C	<	92	5C	\	124	7C	
29	1D	.	61	3D	=	93	5D]	125	7D	}
30	1E	.	62	3E	>	94	5E	^	126	7E	~
31	1F	.	63	3F	?	95	5F	_	127	7F	.

A. Beck

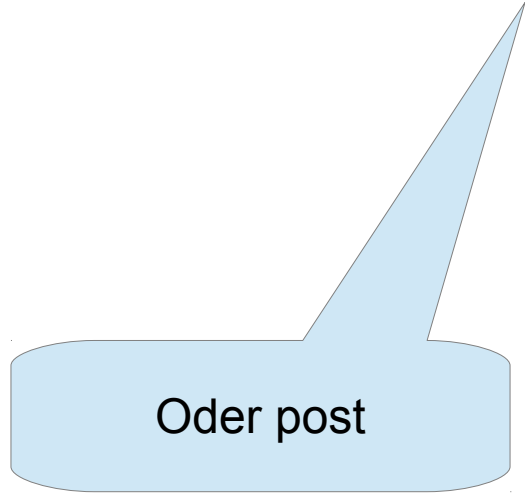
Formulare

Doku unter

<http://de.selfhtml.org/html/formulare/definieren.htm>

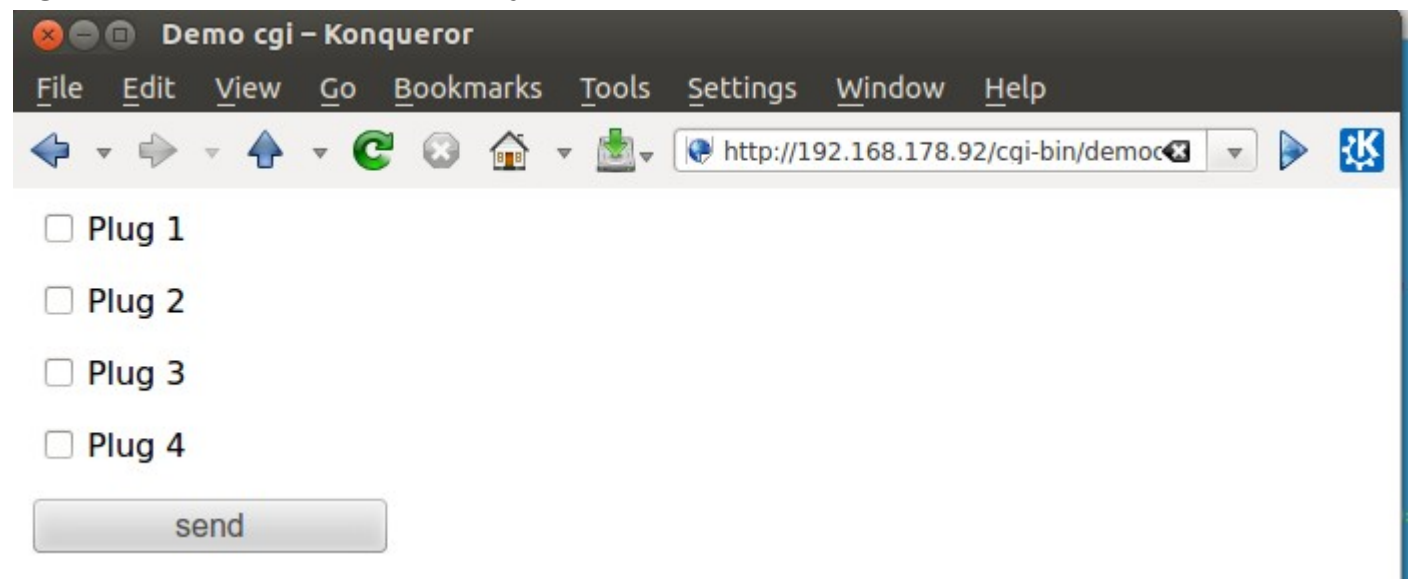
```
<form action="http://myhost/cgi-bin/myprog.cgi" method="get">  
<!-- hier folgen die Formularelemente -->
```

```
</form>
```



Oder post

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
<Title>Demo cgi</Title>
<body>
<form action="http://192.168.178.92/cgi-bin/democgi.cgi" METHOD="POST">
<P><INPUT TYPE=CHECKBOX NAME="check1" VALUE=""
  STYLE=" font-family: 'Arial', sans-serif; font-size: 12pt">Plug 1 </P>
<P><INPUT TYPE=CHECKBOX NAME="check2" VALUE=""
  STYLE=" font-family: 'Arial', sans-serif; font-size: 12pt">Plug 2 </P>
<P><INPUT TYPE=CHECKBOX NAME="check3" VALUE=""
  STYLE=" font-family: 'Arial', sans-serif; font-size: 12pt">Plug 3 </P>
<P><INPUT TYPE=CHECKBOX NAME="check4" VALUE=""
  STYLE=" font-family: 'Arial', sans-serif; font-size: 12pt">Plug 4 </P>
<P><INPUT TYPE="submit" NAME="Submit" VALUE="send"
  STYLE="width: 4.74cm; height: 0.77cm; font-family: 'Arial', sans-serif; font-size: 12pt"></P>
</form>
</body>
</html>
```



Parameterübergaben

Querystring:

check1=&check2=&Submit=send

& trennt die Controls

= Wert des Controls, falls sinnvoll

☒ Plug 1

☒ Plug 2

☐ Plug 3

☐ Plug 4

send

Mit Funktionen aus `string.h` kann der Querystring untersucht werden

```
if(strstr(Param, "check1" ) ) ...
```


Ausgabe von html

- Start mit

```
printf("Content-Type: text/html");  
printf("\n\n") ;
```

- Ausgabe von html-Text aus html-Datei

```
F=fopen("/var/www/demo.html","rt");  
while (fgets(buf,128,F))  
{  
    . . .  
    puts(buf);  
}  
fclose(F);
```

Hier variable Inhalte einpflegen
und html-Text programmtechnisch
erzeugen

Dynamisch content einbauen

```
while (fgets(buf,128,F))
{
    if (strstr(buf,"check1"))
    {
        p=strstr(buf,"VALUE=");
        if(strstr(Param,"check1"))
        {
            cbstat|= 1; strcpy(p+strlen("VALUE=\"\""), "CHECKED");
        }
        else
        {
            cbstat&=~1; strcpy(p+strlen("VALUE=\"\""), " ");
        }
        . . .
        puts(buf);
    }
}
```

html-Zeile mit
Checkbox check1
gefunden

War check1 angeklickt?

Status berechnen

Häckchen

No Häckchen

Licht ein-/ausschalten

```
for (i=0;i<2;i++)  
{  
    if (cbstat&(1<<i))  
        sprintf(command, "/usr/local/bin/send 11011 %d 1", i+1);  
    else  
        sprintf(command, "/usr/local/bin/send 11011 %d 0", i+1);  
    system(command);  
}
```

