
HTM2COB 1.00

User's Guide

HTML to COBOL converter

László Erdős, email: lex-x@freenet.de

Facebook: <https://www.facebook.com/wortfee>

HTM2COB (and this document) is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

HTM2COB (and this document) is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with HTM2COB (and this document).

If not, see <http://www.gnu.org/licenses>

Summary of Changes

Date	Edition	Change Description
2019.05.01	v01.00	Initial release of this document.

Table of Contents

1. Introduction.....	5
1.1 What does it do?.....	5
1.2 Installation and Configuration.....	5
1.2.1 Directory structure.....	5
1.2.2 Compile and run.....	5
1.3 Test.....	6
1.4 Requirements.....	6
1.5 Features of the generated CGI server.....	6
2. Usage.....	7
2.1 Command line options of HTM2COB.....	7
2.2 A detailed example.....	7
2.2.1 Input file name convention.....	7
2.2.2 Embedding COBOL code in the HTML file.....	7
2.2.3 Run the HTM2COB converter.....	8
2.3 The parameter file.....	9
2.4 Built-in modules.....	9
2.4.1 HTM2COB-POST.....	9
2.4.2 HTM2COB-SPEC-CHARS.....	9
2.4.3 HTM2COB-ENV.....	9
2.4.4 HTM2COB-POST-MULTI.....	9
2.4.5 HTM2COB-SESSION-ID.....	10
2.4.6 HTM2COB-SESSION.....	10
2.4.7 HTM2COB-SET-COOKIE.....	10
2.4.8 HTM2COB-GET-COOKIE.....	11
2.4.9 HTM2COB-UPD-SESS-STR.....	11
2.4.10 imgcaptcha.....	11
2.4.11 imgscale.....	11
2.5 Some security hints.....	11
3. Examples.....	14
3.1 Samples.....	14
3.1.1 0001_hello_get.....	14
3.1.2 0002_hello_post.....	16
3.1.3 0003_hello_form_get.....	17
3.1.4 0004_hello_form_post.....	19
3.1.5 0005_embedding.....	20
3.1.6 0006_comment.....	21
3.1.7 0007_variables.....	22
3.1.8 0008_copyfile.....	24
3.1.9 0009_section.....	25
3.1.10 0010_call.....	28
3.1.11 0011_form_elements.....	30
3.1.12 0012_env_vars.....	32
3.1.13 0013_img_upload.....	34
3.1.14 0014_img_upload_scale.....	36
3.1.15 0015_multiple_menu.....	38
3.1.16 0016_js_alert.....	39

3.1.17	0017_more_button.....	41
3.1.18	0018_link_submit.....	43
3.1.19	0019_menu.....	44
3.1.20	0020_sess_id.....	46
3.1.21	0021_sess_var.....	47
3.1.22	0022_sess_counter.....	48
3.1.23	0023_sess_login.....	50
3.1.24	0024_sess_captcha.....	54
3.1.25	0025_sess_cart.....	58
3.1.26	0026_sess_regen_login.....	65
3.1.27	0027_sess_regen_cart.....	71
3.1.28	0028_ck_visit.....	78
3.1.29	0029_ck_save_data.....	80
3.1.30	0030_ck_param.....	80
3.1.31	0031_ck_sess_login.....	80
3.1.32	0032_ck_sess_captcha.....	80
3.1.33	0033_ck_sess_cart.....	80
3.1.34	0034_ck_sess_regen_login.....	80
3.1.35	0035_ck_sess_regen_cart.....	80
3.1.36	0036_ajax_hello.....	80
3.1.37	0037_ajax_query_str.....	80
3.1.38	0038_jq_form_validate.....	80
3.1.39	0039_jq_jcarousel.....	80
3.2	Projects.....	81
4.	Bibliography and references.....	82

1. Introduction

1.1 What does it do?

With the HTM2COB tool you can convert an HTML file into COBOL source code. The generated COBOL program is a CGI server program. Into the HTML input you can additionally embed COBOL codes with a simple syntax. After generating the COBOL program, you can immediately compile and install it onto your web server.

1.2 Installation and Configuration

1.2.1 Directory structure

htm2cob	→ main directory
htm2cob/readme.txt	→ readme file
htm2cob/doc/HTM2COB_v01_0.odt	→ this document in LibreOffice format
htm2cob/doc/HTM2COB_v01_0.pdf	→ this document in PDF
htm2cob/examples/projects	→ project examples (this is a TO DO)
htm2cob/examples/samples	→ sample examples (see below)
htm2cob/source/htm2cob.cob	→ the HTM2COB converter program
htm2cob/source/htm2cob0.cpy	→ template: WORKING-STORAGE
htm2cob/source/htm2cob1.cpy	→ template: PROCEDURE DIVISION
htm2cob/source/htm2cob2.cpy	→ template: CGI fields / values
htm2cob/source/htm2cob3.cpy	→ template: CGI environment variables
htm2cob/source/htm2cob4.cpy	→ template: decode UTF-8 chars
htm2cob/source/htm2cob5.cpy	→ template: replace HTML special chars
htm2cob/source/htm2cob6.cpy	→ template: session ID with SHA3-512
htm2cob/source/htm2cob7.cpy	→ template: session file handling
htm2cob/source/htm2cob8.cpy	→ template: create cookie string
htm2cob/source/Makefile	→ make file for the converter

In the samples folder are some examples. All examples have a separated make file and a screenshot directory.

1.2.2 Compile and run

First compile the HTM2COB converter under htm2cob/source/ and after it the examples. The examples use the converter. The provided make files are for cygwin. For other environments you have to do some small changes in the make files. For example you have to customize the target directories for your web server.

1.3 Test

This program was developed and tested using:

- Windows 10 (64 bit) running on a HP laptop
- GnuCOBOL 3.1-dev.0, built on Aug 17 2019
- cygwin (64 bit)
- Firefox Quantum 66.0.3 (64-Bit)

1.4 Requirements

- To run the examples you need a web server.
- If you are using GnuCOBOL with BDB (Berkeley DB) you have to set the environment variable DB_HOME to where your session files are saved.
- HTM2COB includes the cobsha3 GnuCOBOL contribution. The cobsha3 GnuCOBOL contribution was developed and tested in a little-endian environment. There are some issues with big-endian, because of redefines and BINARY-DOUBLE. **Very important:** Before you use HTM2COB, you have to run the test for cobsha3 program in your environment, and then you have to check the results!

1.5 Features of the generated CGI server

- HTTP request methods: GET, POST, POST with image upload.
- Debug flag for testing.
- It uses utf-8.
- Definable max data length parameters.
- Image upload options: BMP, GIF, JPG, PNG, TIF.
- Session ID generation with SHA3-512.
- Session file handling. Read / Write additional data for the session ID.
- Time out parameters for the session ID.
- Cookie string creation with formatted time stamp.
- Setting and getting cookies.

2. Usage

2.1 Command line options of HTM2COB

htm2cob <input HTML file> [-m] [-v]

htm2cob converts a HTML file in a CGI COBOL program

Options:

<input HTML file>	This is a HTML file with embedded COBOL
-m, -module	It converts only the lines in <BODY>
-v, -verbose	Verbose mode

2.2 A detailed example

2.2.1 Input file name convention

In addition to the HTML file, the converter also needs a parameter file, which is a COBOL copy file. Both file names have a special format. The input HTML file has the suffix "_cob.html" and the parameter file has the suffix "_param.cpy". The prefix must be the same. The converter will change the suffix "_cob.html" to ".cob" in the generated output file.

Example:

Input files: 0004_hello_form_post_cob.html
 0004_hello_form_post_param.cpy

Output file: 0004_hello_form_post.cob

2.2.2 Embedding COBOL code in the HTML file

There are only two identifiers for embedding the COBOL code in HTML. One for the variable definitions, and one for program codes.

Defining variables: <?cob-ws ... ?>

Defining program codes: <?cob ... ?>

It does not matter where you define the variables, the converter will place them all in the WORKING-STORAGE SECTION.

An example for variable definition:

```
<?cob-ws
*> You can define variables in body or outside of body also.
01 fname                pic x(100) .
01 lname                pic x(100) .
01 htm-fname            pic x(100) .
01 htm-lname            pic x(100) .
?>
```

An example for program code:

```
<?cob display "Hello GnuCOBOL world!" end-display ?>
```

Everything up to "</html>" from the HTML file is placed in the main section in the generated output file. You can define your own sections after the "</html>". These sections are placed after the main section in the generated output file.

2.2.3 Run the HTM2COB converter

```
$ make
../../source/htm2cob 0004_hello_form_post_cob.html -v
Input file: 0004_hello_form_post_cob.html
Output file: 0004_hello_form_post.cob
PROGRAM-ID. 0004_hello_form_post.
```

Count	Line	Col	Tag
1	3	1	<HTML>
2	9	1	<BODY>
3	11	1	<?COB
4	16	1	?>
5	19	1	<?COB
6	19	51	?>
7	22	13	<?COB
8	22	55	?>
9	23	13	<?COB
10	23	55	?>
11	26	1	</BODY>
12	27	1	</HTML>
13	29	1	<?COB-WS
14	35	1	?>

Check tags...

```
cobc -x -free 0004_hello_form_post.cob -o 0004_hello_form_post
cp 0004_hello_form_post /srv/www/cgi-bin/0004_hello_form_post
cp 0004_hello_form_post_test.html
/srv/www/htdocs/0004_hello_form_post_test.html
```

The converter always checks the tags. With the verbose parameter "-v" you can see the begin and end tags in the output list.

There are cases when you need multiple end BODY and end HTML, e.g. after a possible abort because of errors. The converter can not handle these cases, so you have to code them differently.

You can see this in the example 0023_sess_login_2_cob.html:

```
*>      slash with concatenate, because of the htm2cob preprocessor
      DISPLAY "</" "body">" "</" "html">"
```

2.3 The parameter file

The parameter file has the suffix "_param.cpy". It is well commented, and always has the same structure. It is important that you set the parameters carefully. It is security relevant. Set the right request method, number of input fields, memory and other options, what you really need in your server.

2.4 Built-in modules

There are some built-in modules in the generated COBOL program. Here is a short list, but you can see the usage of these modules in the examples later.

2.4.1 HTM2COB-POST

The CGI field names and values will be saved in a memory table and in a variable. With this module you can read the value for a given name. (You can also access the HTM2COB-TABLE and the HTM2COB-DATA-VALUE from your program, if you want. See the generated COBOL code.)

Example:

```
call "HTM2COB-POST" using "firstname", fname end-call
```

2.4.2 HTM2COB-SPEC-CHARS

This module replaces HTML special chars. You have to call it always before writing something back to the client.

Example:

```
call "HTM2COB-SPEC-CHARS" using fname, htm-fname end-call
```

2.4.3 HTM2COB-ENV

This module reads an environment variable.

Example:

```
CALL "HTM2COB-ENV" USING LNK-HTM2COB-ENV END-CALL
```

2.4.4 HTM2COB-POST-MULTI

With a "multiple selection menu" you can get the same CGI variable name multiple times in the internal table. With this module you can read these. (See

10

later in example 0015_multiple_menu.)

Example:

```
*> use ; as a separator char
    call "HTM2COB-POST-MULTI" using "pizza", ";", ws-pizza end-call
```

2.4.5 HTM2COB-SESSION-ID

This module generates a session ID with the SHA3-512 algorithm. As input the REMOTE-ADDR and HTTP-USER-AGENT will be used. (See below in example 0020_sess_id.)

Example:

```
CALL "HTM2COB-SESSION-ID" USING LNK-HTM2COB-SESSION-ID END-CALL
```

2.4.6 HTM2COB-SESSION

This module writes or deletes the session data in a file. You can call it directly or thru Sections. (See later in example 0021_sess_var, and in other session examples.)

Example:

```
CALL "HTM2COB-SESSION" USING LNK-HTM2COB-SESSION END-CALL
```

Example with Sections:

```
*> start or resume session
    PERFORM HTM2COB-SESS-START

*> delete old sessions
    PERFORM HTM2COB-SESS-DEL-OLD

*> delete a session
    PERFORM HTM2COB-SESS-DESTROY

*> regenerate a session (create a new session, copy old session data in
*> new session, delete old session)
    PERFORM HTM2COB-SESS-REGENERATE

*> set session var
    MOVE "mydog" TO HTM2COB-SESS-VAR-NAME
    MOVE "Morgo" TO HTM2COB-SESS-VAR-VALUE
    PERFORM HTM2COB-SESS-SET

*> get session var
    MOVE "mydog" TO HTM2COB-SESS-VAR-NAME
    PERFORM HTM2COB-SESS-GET

*> del session var
    MOVE "mydog" TO HTM2COB-SESS-VAR-NAME
    PERFORM HTM2COB-SESS-DEL
```

2.4.7 HTM2COB-SET-COOKIE

This module creates a cookie string. (See later in example 0026_cookie_visit and

in other cookie examples.)

Example:

```
CALL "HTM2COB-SET-COOKIE" USING LNK-HTM2COB-SET-COOKIE END-CALL
```

2.4.8 HTM2COB-GET-COOKIE

This module reads a saved cookie value from the HTM2COB-HTTP-COOKIE field. (See later in example 0026_cookie_visit and in other cookie examples.)

Example:

```
CALL "HTM2COB-GET-COOKIE" USING LNK-HTM2COB-GET-COOKIE END-CALL
```

2.4.9 HTM2COB-UPD-SESS-STR

This module updates SESSIONID in the HTM2COB-HTTP-COOKIE field.

2.4.10 imgcaptcha

This is a separate C module. It creates a PNG image captcha file. (See later in example 0024_sess_captcha.)

Example:

```
CALL "imgcaptcha" USING BY REFERENCE WS-CAPTCHA-FILE-NAME
                        , BY REFERENCE WS-CAPTCHA-CHAR-TYPE
                        , BY REFERENCE WS-CAPTCHA-TEXT
RETURNING WS-RETVAL
END-CALL
```

2.4.11 imgscale

This is a separate C module. It converts and scales an image. (See below in example 0014_img_upload_scale.)

Example:

```
CALL "imgscale" USING BY REFERENCE WS-IMG-IN
                    , BY REFERENCE WS-IMG-IN-TYPE
                    , BY REFERENCE WS-IMG-OUT
                    , BY REFERENCE WS-IMG-OUT-TYPE
                    , BY REFERENCE WS-WIDTH-OUT
RETURNING WS-RETVAL
END-CALL
```

2.5 Some security hints

Web development has to deal a lot more than traditional programming with security risks. Your application will be accessible to the whole world, and this includes hackers. **You have to stay up to date with potential security risks.** Read literature on secure web development. If you have a lack of knowledge in this field then don't change things that you don't know, the effects on the privacy could be severe.

- Always use HTTPS and not HTTP on your production web server. Hypertext Transfer Protocol Secure (HTTPS) is an extension of the

Hypertext Transfer Protocol (HTTP) for secure, encrypted communication over a computer network: <https://en.wikipedia.org/wiki/HTTPS>. Free SSL certificates can be acquired from the Let's Encrypt initiative.

- Never enable warning or with error messages on your production web server since they could aid hackers or show parts of your program that have to remain unknown. Set the flag HTM2COB-DEBUG-FLAG on V-HTM2COB-DEBUG-NO and check all "display" in your COBOL code.
- It is important for security reasons that you set the parameters carefully. Set the right request method, number of fields, bytes of maximum memory and other options, depending on your needs in your server program. For example, if you do not need a file upload, or session handling, then do not allow these. Always set the maximum allowed size for fields and upload files. For images set the allowed image type.
- If possible, do not use the GET request method. GET requests include all required data in the browser URL which could leave sensitive data exposed to hackers. Instead of GET, you should use POST.
- You have to validate every data the client returns. A client side validation is not enough.
- Make sure that your application handles cases where POST or GET data is expected but not returned by your user (or a hacker).
- Use the "HTM2COB-SPEC-CHARS" filter before you write a text back. If you get something from the client, and you write it back later, then you should filter it to prevent cross site scripting (https://en.wikipedia.org/wiki/Cross-site_scripting).
- When you set a cookie, you have to specify the domain and subdomain of your application.
- Always use cookies with the "HttpOnly" parameter. This prevents javascript functions from manipulating the cookies you set.
- Use session-cookies, and do not save any other data in cookies, only the session-ID. Data stored in cookies is not secure.
- Cookie data from the client must also be validated, escaped and filtered. If you only store the session-id in the cookie then a hacker might have a harder time.
- Replace all Line-Feed and Carriage-Return chars in your cookie text before you set a cookie. (You should not use any special chars in cookie text, but maybe a client sends you special chars in a text, that you want to use later in cookie.)
- There are many ways to track users, see <https://en.wikipedia.org/wiki/Evercookie>
- **SQL injections are not filtered in this program!!** See https://en.wikipedia.org/wiki/SQL_injection . You have to use embedded SQL or Prepared SQL Statements.

- You MUST NOT store private files under the htdocs folders. These folders are directly accessible from the web. Examples of what NOT to store here: Session files, Data files for your applications, files uploaded by users and so on.
- You should prevent users from loading files higher directories than the one you are working in. Ensure that “..” expressions are filtered, else hackers might even steal your server’s passwd file by simply accessing “/../../etc/passwd”
- You have to set the proper permissions on each directory for access control.
- If you implement a search feature, then make sure that it filters out files that should not be accessible to your user.
- It’s best to regenerate the session-id every time the user visits a new page. If however you have lots of users this might be too much of burden on your server, then you can choose to generate a new session-id only after each n-th visit of your user. The minimum count of session-id generation you can do is one, when the user logs in. If you don’t regenerate the session-id, then hackers could use Session hijacking to log in: https://en.wikipedia.org/wiki/Session_hijacking
- The example programs here store the session files in the /tmp folder. **This is forbidden on any production system!** Session files are sensitive and should be private, the /tmp folder is accessible to any user of your server!
- Every separate application should use it’s own session files. The parameter files specify where the session files of the application are located.
- In case of unsuccessful logins, the login form must not give details about which of the entries was incorrect. This is also true for the user registration (account creation) form.
- Choose a sensible minimum length requirement for the passwords, and verify it on the server side. Require users to use lower and upper case characters, numbers and special characters.
- Do not store user input data from the outside world in any log files. Hackers could get entry by submitting special entries.
- Use salting and peppering for your login forms:
[https://en.wikipedia.org/wiki/Salt_\(cryptography\)](https://en.wikipedia.org/wiki/Salt_(cryptography))
[https://en.wikipedia.org/wiki/Pepper_\(cryptography\)](https://en.wikipedia.org/wiki/Pepper_(cryptography))

Never save your user’s password in clear text!! This includes not saving it in a file or in a database. Even the RAM is sensitive to attacks, so make sure to initialize the variable where the user’s clear-text password is stored after you generated the hash for it.

You **have** to use a hashing algorithm to ensure the security of your users.

3. Examples

3.1 Samples

3.1.1 0001_hello_get

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Hello GnuCOBOL world!</title>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<meta http-equiv="Content-Style-Type" content="text/css" />
</head>
<body>
This is a GET without values.<br><br>
<?cob display "Hello GnuCOBOL world!" end-display ?>
</body>
</html>

```

HTML with embedded COBOL

Get request method without values.

```

*> ***** begin user defined content HTM2COB-MAIN SECTION *****

DISPLAY "<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"" END-DISPLAY
DISPLAY "    "http://www.w3.org/TR/html4/loose.dtd">" END-DISPLAY
DISPLAY "<html>" END-DISPLAY
DISPLAY "<head>" END-DISPLAY
DISPLAY "<title>Hello GnuCOBOL world!</title>" END-DISPLAY
DISPLAY "<meta http-equiv="content-type" content="text/html; charset=utf-8" />" END-DISPLAY
DISPLAY "<meta http-equiv="Content-Style-Type" content="text/css" />" END-DISPLAY
DISPLAY "</head>" END-DISPLAY
DISPLAY "<body>" END-DISPLAY
DISPLAY "This is a GET without values.<br><br>" END-DISPLAY

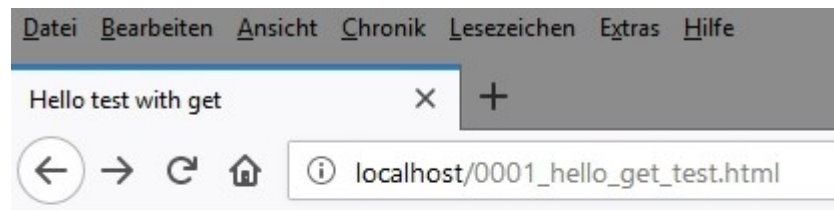
display "Hello GnuCOBOL world!" end-display

DISPLAY "</body>" END-DISPLAY
DISPLAY "</html>" END-DISPLAY

*> ***** end user defined content HTM2COB-MAIN SECTION *****

```

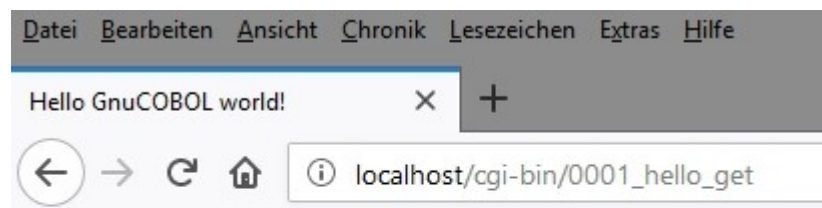
Generated COBOL code



Hello test with get

Send

0001_hello_get_test



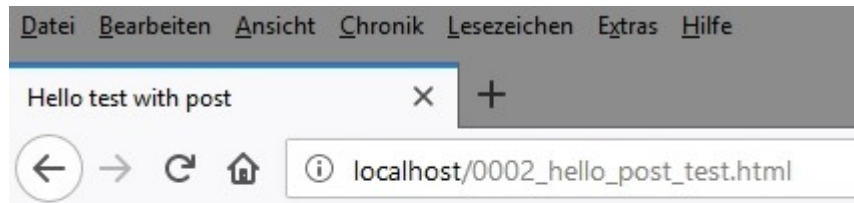
This is a GET without values.

Hello GnuCOBOL world!

0001_hello_get

3.1.2 0002_hello_post

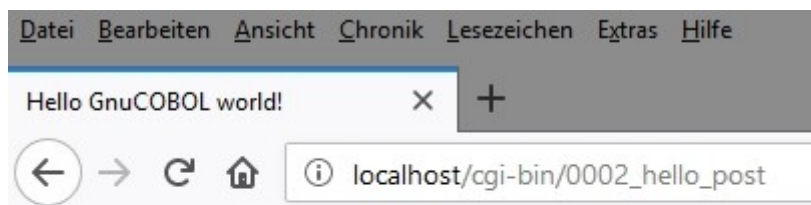
Post request method without values.



Hello test with post

Send

0002_hello_post_test



This is a POST without values.

Hello GnuCOBOL world!

0002_hello_post

3.1.3 0003_hello_form_get

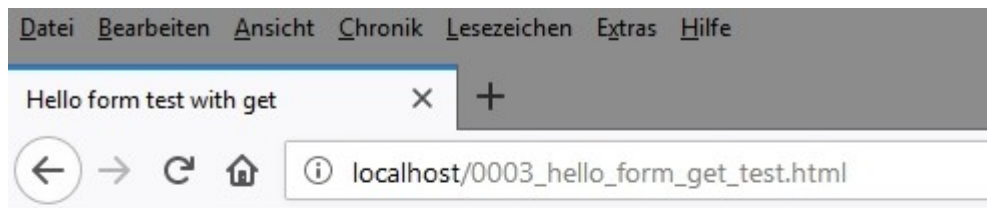
GET request method with values. If it possible, do not use the GET request method. **GET requests include all data in the browser URL**. Instead of GET, you should use POST because of security reasons.

This is also an example of usage of the modules "HTM2COB-POST" and "HTM2COB-SPEC-CHARS". You have to call "HTM2COB-SPEC-CHARS" always, before writing something back to the client.

```
<?cob
    call "HTM2COB-POST" using "firstname", fname end-call
    call "HTM2COB-SPEC-CHARS" using fname, htm-fname end-call
    call "HTM2COB-POST" using "lastname", lname end-call
    call "HTM2COB-SPEC-CHARS" using lname, htm-lname end-call
?>
This is a GET with values.<br><br>

<?cob display "Hello GnuCOBOL world!" end-display ?>
|
<p>
First name: <?cob display trim(htm-fname) end-display ?><br>
Last name : <?cob display trim(htm-lname) end-display ?><br>
</p>
```

HTML with embedded COBOL

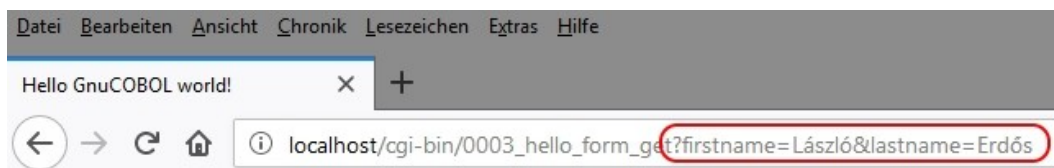


Hello form test with get

First name:

Last name:

0003_hello_form_get_test



This is a GET with values.

Hello GnuCOBOL world!

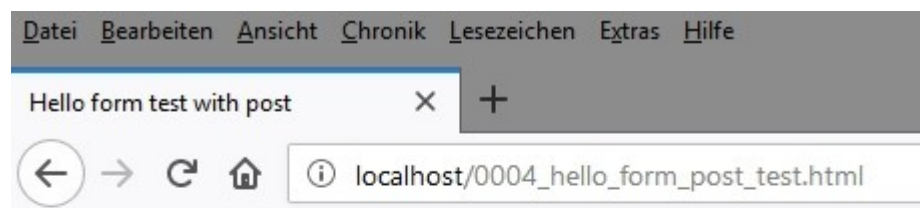
First name: László
Last name : Erdős

0003_hello_form_get

3.1.4 0004_hello_form_post

POST request method with values.

This also is an example of using the modules "HTM2COB-POST" and "HTM2COB-SPEC-CHARS". You have to call "HTM2COB-SPEC-CHARS" always, before writing something back to the client.

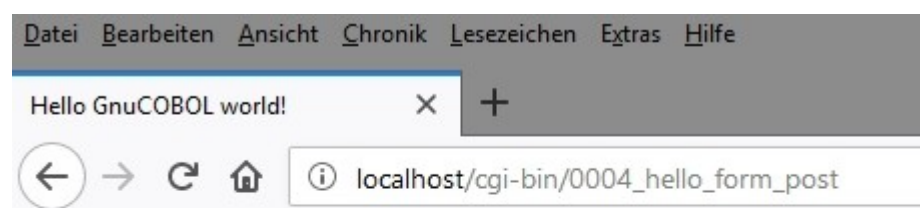


Hello form test with post

First name:

Last name:

0004_hello_form_post_test



This is a POST with values.

Hello GnuCOBOL world!

First name: László
Last name : Erdős

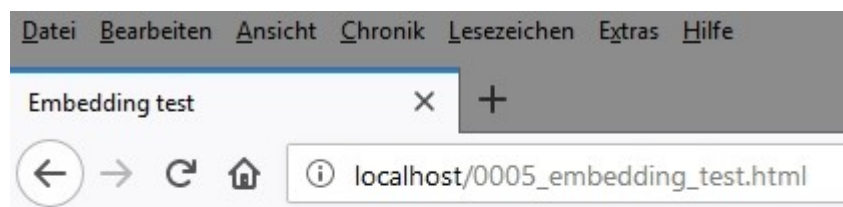
0004_hello_form_post

3.1.5 0005_embedding

Example of embedding COBOL lines in HTML lines.

```
<body>
The first line in HTML<br/>
<?cob display "The second line in COBOL<br/>" ?>
The third line in HTML<br/>
<?cob
    display "The fourth line in COBOL<br/>"
    display "The fifth line in COBOL"
?>
</body>
```

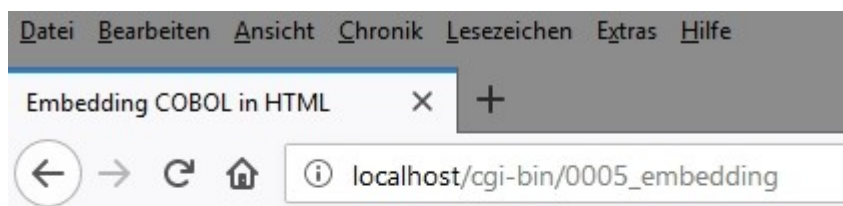
HTML with embedded COBOL



Embedding test

Send

0005_embedding_test



The first line in HTML
The second line in COBOL
The third line in HTML
The fourth line in COBOL
The fifth line in COBOL

0005_embedding

3.1.6 0006_comment

Example of COBOL comments in the embedded COBOL.

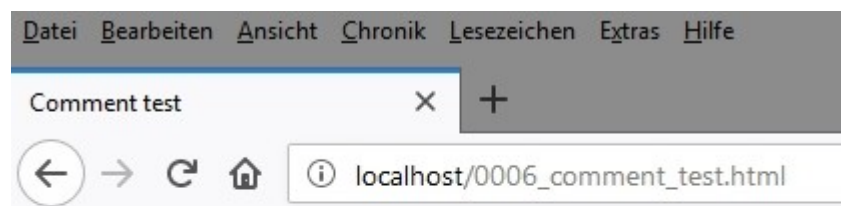
```
<body>
<?cob-ws
*> This is a COBOL comment in column 1.
01 text-1 pic x(30) value "Welcome in Hotel Moon!".
01 price pic 9(3).
01 price-disp pic ZZ9. *> comment to end of line
?>

<?cob-ws*> This is a COBOL comment in column 1. ?>

<?cob display "This is the start of the program.<br>" *> comment to end of line ?>
<?cob
*> This is a COBOL comment in column 1.
display "This is the end of the program.<br>"
*> comment to end of line
?>

<?cob*> This is a COBOL comment in column 1. ?>
</body>
```

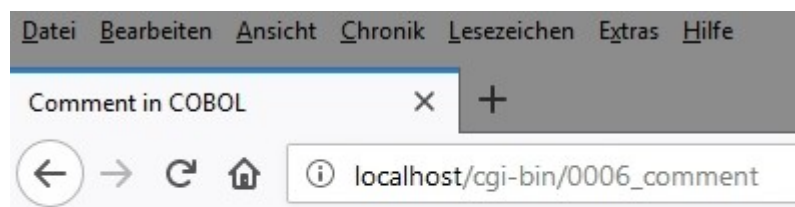
HTML with embedded COBOL



Comment test

Send

0006_comment_test



This is the start of the program.
This is the end of the program.

0006_comment

3.1.7 0007_variables

You can define your variables everywhere with "<?cob-ws", they will be copied in WORKING-STORAGE SECTION.

```
<body>
<?cob-ws
*> You can define your variables everywhere,
*> they will be copied in WORKING-STORAGE SECTION.
  01 text-1 pic x(30) value "Welcome in Hotel Triton!".
  01 price  pic 9(3).
  01 price-disp pic ZZ9.
?>

<?cob compute price = room + breakfast
  move price to price-disp ?>

<?cob-ws 01 room pic 9(3) value 50. ?>

<h1><font color=#"FF0000"><?cob display text-1 ?></font></h1>
<p><h2>
Price with breakfast: <?cob display price-disp ?> Euro
</h2></p>
<hr color=#"00FF00">

<?cob-ws 01 breakfast pic 9(3) value 8. ?>

</body>
```

HTML with embedded COBOL

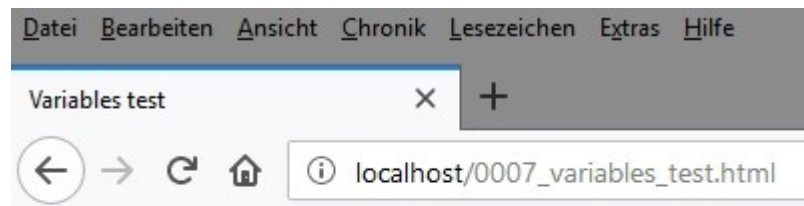
```
*> ***** begin user defined content WORKING-STORAGE SECTION *****

*> You can define your variables everywhere,
*> they will be copied in WORKING-STORAGE SECTION.
  01 text-1 pic x(30) value "Welcome in Hotel Triton!".
  01 price  pic 9(3).
  01 price-disp pic ZZ9.

  01 room pic 9(3) value 50.
01 breakfast pic 9(3) value 8.

*> ***** end user defined content  WORKING-STORAGE SECTION *****
```

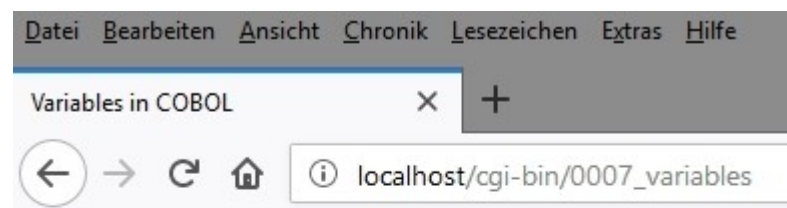
Generated COBOL code



Variables test

Send

0007_variables_test



Welcome in Hotel Triton!

Price with breakfast: 58 Euro

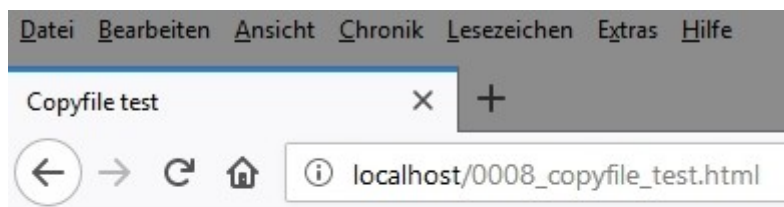
0007_variables

3.1.8 0008_copyfile

You can use copy files.

```
<?cob-ws
*> you can use copy files also
copy "cobcopy.cpy".
?>
```

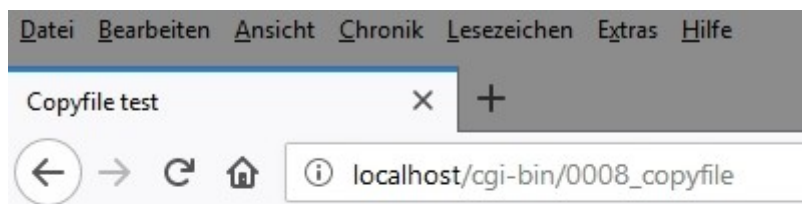
HTML with embedded COBOL



Copyfile test

Send

0008_copyfile_test



Welcome in Hotel Moon!

Price with breakfast: 42 Euro

0008_copyfile

3.1.9 0009_section

COBOL sections after the HTML body will be copied after the HTM2COB-MAIN section in the generated COBOL code.

```
<?cob
*> COBOL sections after html body will be copied after the HTM2COB-MAIN
*> section in the generated COBOL code.
LIST-BOOK SECTION.

    PERFORM VARYING WS-IND FROM 1 BY 1
        UNTIL WS-IND > C-MAX-IND

    INITIALIZE WS-TAB-FIELDS
    UNSTRING WS-BOOKS-LINE(WS-IND) DELIMITED BY ";"
        INTO WS-SALES
            WS-TITLE
            WS-AUTHOR

    DISPLAY
        "<tr>"
        "<td>"WS-IND"</td>"
        "<td>"WS-SALES"</td>"
        "<td>"WS-TITLE"</td>"
        "<td>"WS-AUTHOR"</td>"
        "</tr>"

    END-DISPLAY
END-PERFORM

EXIT SECTION .
?>
```

Section outside of HTML body.

```
*> COBOL sections after html body will be copied after the HTM2COB-MAIN
*> section in the generated COBOL code.
```

```
LIST-BOOK SECTION.
```

```
PERFORM VARYING WS-IND FROM 1 BY 1
      UNTIL WS-IND > C-MAX-IND

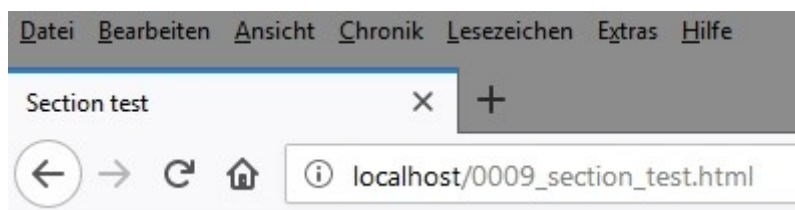
INITIALIZE WS-TAB-FIELDS
UNSTRING WS-BOOKS-LINE(WS-IND) DELIMITED BY ";"
      INTO WS-SALES
      WS-TITLE
      WS-AUTHOR

DISPLAY
  "<tr>"
  "<td>"WS-IND"</td>"
  "<td>"WS-SALES"</td>"
  "<td>"WS-TITLE"</td>"
  "<td>"WS-AUTHOR"</td>"
  "</tr>"

END-DISPLAY
END-PERFORM

EXIT SECTION .
```

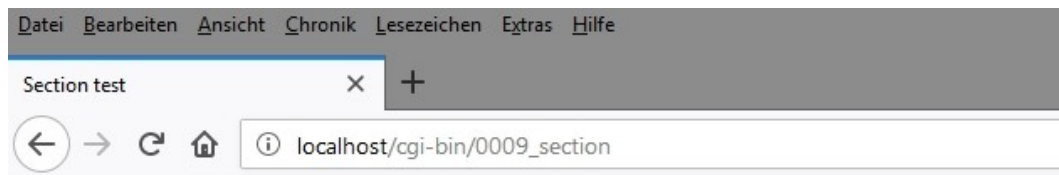
Generated COBOL code



Section test

Send

0009_section_test



Top 10 Most Read Books In The World

Nr.	Approximate sales	Title	Author
01	3.9 Billion Copies	The Bible	---
02	820 Million Copies	Quotations from the Works of Mao Tse-tung	By Mao Tse-tung
03	400 Million Copies	Harry Potter	By J. K. Rowling
04	103 Million Copies	Lord of the Rings	By J. R. R. Tolkien
05	65 Million Copies	The Alchemist	By Paulo Coelho
06	57 Million Copies	The Da Vinci Code	By Dan Brown
07	43 Million Copies	Twilight - The Saga	By Stephenie Meyer
08	33 Million Copies	Gone With the Wind	By Margaret Mitchell
09	30 Million Copies	Think and Grow Rich	By Napoleon Hill
10	27 Million Copies	Diary of Anne Frank	By Anne Frank

0009_section

3.1.10 0010_call

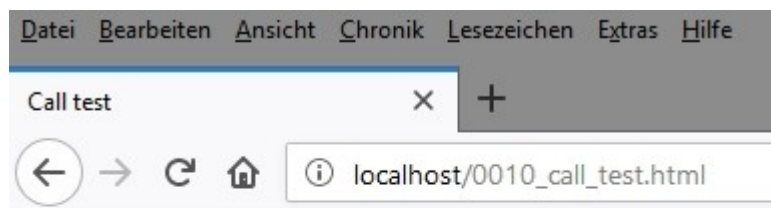
Example for calling COBOL modules.

```
<?cob
    DISPLAY "<h2>Start main</h2>"

    SET SUB-PROG-1 TO TRUE
    CALL SUB-PROG-NAME END-CALL

    SET SUB-PROG-2 TO TRUE
    CALL SUB-PROG-NAME USING TEST-DATA
                                TEST-DATA-LEN
    END-CALL
?>
```

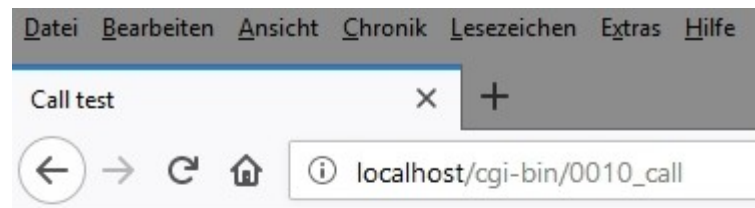
COBOL call



Call test

Send

0010_call_test



Dynamic CALL test

Start main

Start 0010_sub1

Start 0010_sub2

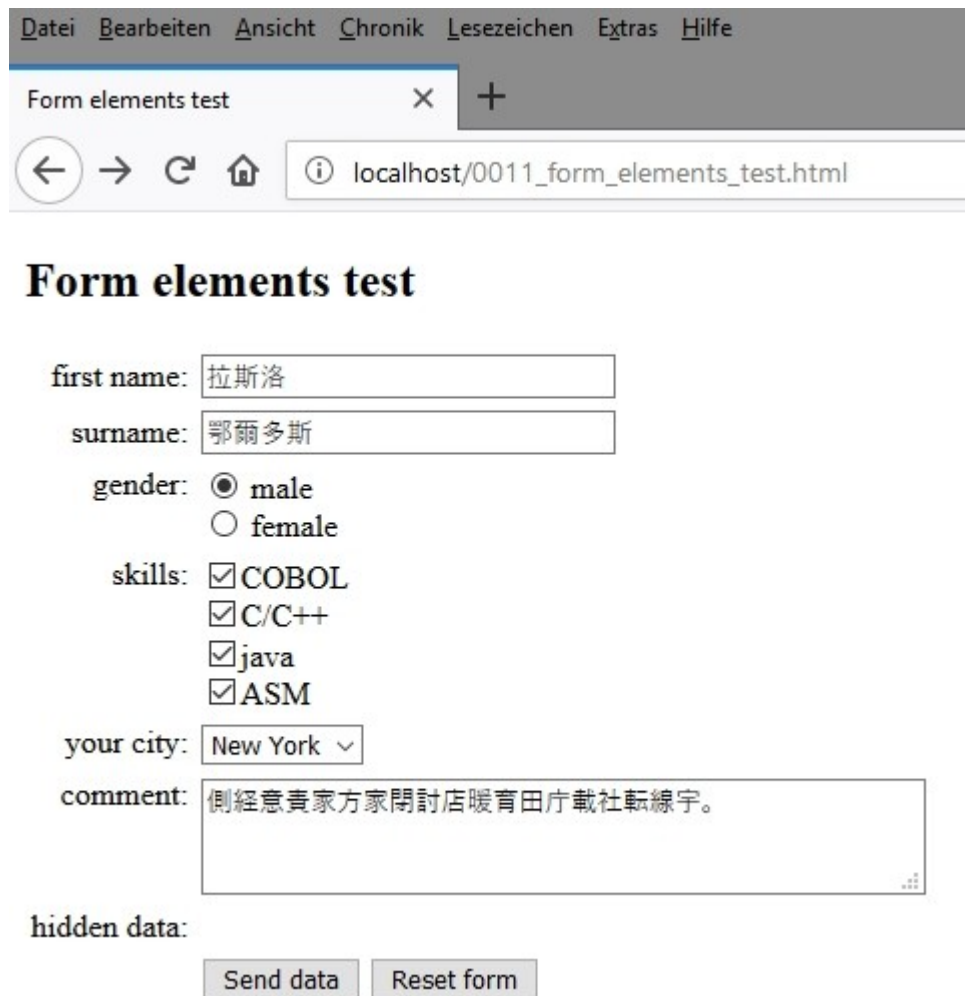
LNK-DATA: abcde12345

LNK-DATA-LEN: +0000000010

0010_call

3.1.11 0011_form_elements

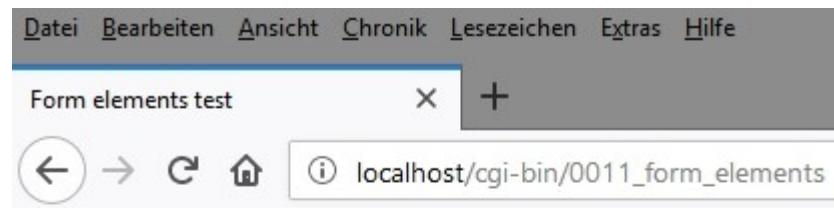
HTML form elements with Chinese characters, a demonstration of the utf-8 support.



The screenshot shows a web browser window with the title 'Form elements test'. The address bar displays 'localhost/0011_form_elements_test.html'. The form contains the following elements:

- first name:** Text input field containing '拉斯洛'.
- surname:** Text input field containing '鄂爾多斯'.
- gender:** Radio buttons for 'male' (selected) and 'female'.
- skills:** Checkboxes for 'COBOL', 'C/C++', 'java', and 'ASM', all of which are checked.
- your city:** Dropdown menu showing 'New York'.
- comment:** Text area containing the text '側經意責家方家閉討店暖育田疋載社軋線宇。'.
- hidden data:** Label with no visible input field.
- Buttons:** 'Send data' and 'Reset form' at the bottom.

0011_form_elements_test



Form elements test

First name : 拉斯洛

Surname : 鄂爾多斯

Gender : m

Skill-1 : cobol

Skill-2 : cpp

Skill-3 : java

Skill-4 : asm

Your city : new_york

Comment : 側經意責家方家閉討店暖育田庁載社転線宇。

Hidden data: 1234567890

0011_form_elements

3.1.12 0012_env_vars

This example prints the CGI Environment Variables. It is also an example of using the module "HTM2COB-ENV"

```
<?cob
LIST-ENV-VARS SECTION.

    PERFORM VARYING WS-IND FROM 1 BY 1
        UNTIL WS-IND > C-MAX-IND

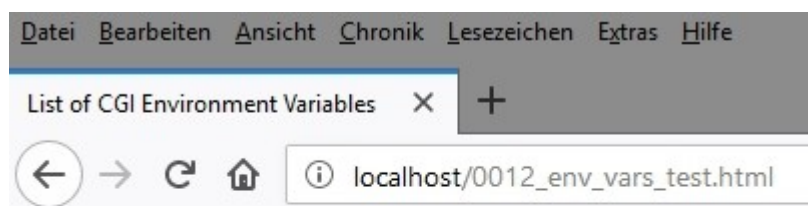
    INITIALIZE WS-TAB-FIELDS
    UNSTRING WS-ENV-LINE(WS-IND) DELIMITED BY ";"
        INTO WS-ENV-NAME
        WS-ENV-DESC

    MOVE FUNCTION TRIM(WS-ENV-NAME) TO LNK-ENV-NAME OF LNK-HTM2COB-ENV
    CALL "HTM2COB-ENV" USING LNK-HTM2COB-ENV END-CALL
    MOVE FUNCTION TRIM(LNK-ENV-VALUE OF LNK-HTM2COB-ENV) TO WS-ENV-TXT

    DISPLAY
        "<tr>"
        "<td>"WS-IND"</td>"
        "<td>"WS-ENV-NAME"</td>"
        "<td>"WS-ENV-DESC"</td>"
        "<td>"WS-ENV-TXT"</td>"
        "</tr>"
    END-DISPLAY
END-PERFORM

EXIT SECTION .
?>
```

Call "HTM2COB-ENV"



List of CGI Environment Variables

Send

0012_env_vars_test



List of CGI Environment Variables

Nr.	Env. Var	Desc.	Value
0001	DOCUMENT_ROOT	The root directory of your server	/srv/www/htdocs
0002	HTTP_COOKIE	The visitor's cookie, if one is set	
0003	HTTP_HOST	The hostname of the page being attempted	localhost
0004	HTTP_REFERER	The URL of the page that called your program	http://localhost/0012_env_vars_test.html
0005	HTTP_USER_AGENT	The browser type of the visitor	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
0006	HTTPS	on, if the program is being called through a secure server	
0007	PATH	The system path your server is running under	/usr/local/bin:/usr/bin:/cygdrive/c/Program Files (x86)/Common Files/Oracle/Java/javapath:/cygdrive/c/ProgramData/Oracle/Java/javapath:/cygdrive/c/Windows/System32:/cygdrive/c/Windows:/cygdrive/c/Windows/System32/wbem:/cygdrive/c/Windows/System32/WindowsPo
0008	QUERY_STRING	The query string	
0009	REMOTE_ADDR	The IP address of the visitor	127.0.0.1
0010	REMOTE_HOST	The hostname of the visitor (if your server has reverse-name-lookups on, otherwise this is the IP address again)	
0011	REMOTE_PORT	The port the visitor is connected to on the web server	56946
0012	REMOTE_USER	The visitor's username (for .htaccess-protected pages)	
0013	REQUEST_METHOD	GET or POST	POST
0014	CONTENT_LENGTH	length of content	0
0015	CONTENT_TYPE	type of content	application/x-www-form-urlencoded
0016	REQUEST_URI	The interpreted pathname of the requested document or CGI (relative to the document root)	/cgi-bin/0012_env_vars
0017	SCRIPT_FILENAME	The full pathname of the current CGI	/srv/www/cgi-bin/0012_env_vars
0018	SCRIPT_NAME	The interpreted pathname of the current CGI (relative to the document root)	/cgi-bin/0012_env_vars
0019	SERVER_ADMIN	The email address for your server's webmaster	you@example.com
0020	SERVER_NAME	Your server's fully qualified domain name (e.g. www.name.com)	localhost
0021	SERVER_PORT	The port number your server is listening on	80
0022	SERVER_SOFTWARE	The server software you're using (e.g. Apache 1.3)	Apache/2.4.29 (Unix)

0012_env_vars

3.1.13 0013_img_upload

Image upload example. You have to set the file path for the uploaded files, and set the allowed file types for image uploads in the parameter file.

0013_img_upload_param.cpy.

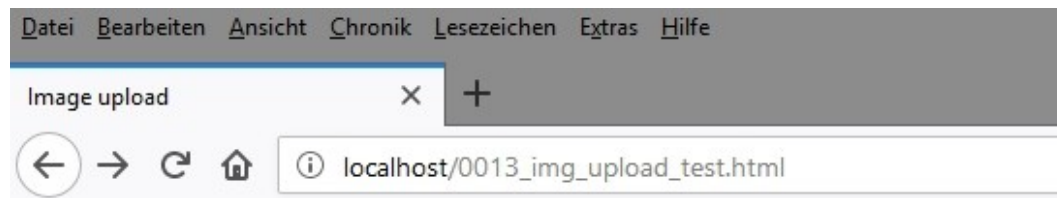


Image upload

For image upload you have to use enctype="multipart/form-data" in form request!

Image description1:

upload1: csabai_kolbasz.jpg

Image description2:

upload2: Trapper.jpg

0013_img_upload_test

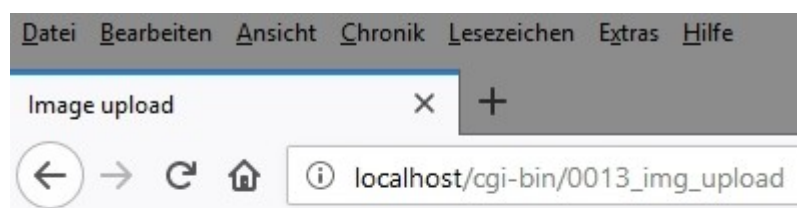
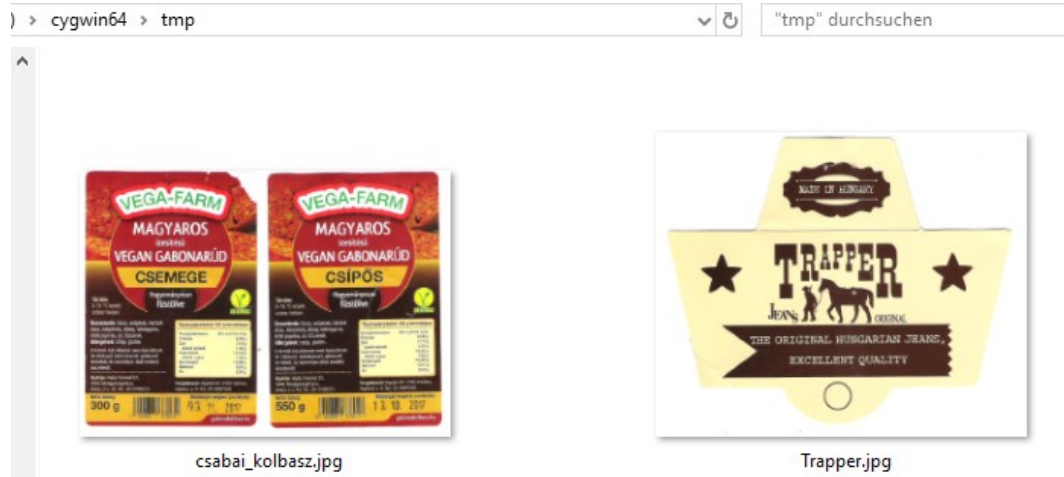


Image upload

Image description1: Vegan csabai kolbasz
 Image upload1: csabai_kolbasz.jpg

Image description2: Trapper, the original jeans
 Image upload2: Trapper.jpg

0013_img_upload



The uploaded images under /tmp

3.1.14 0014_img_upload_scale

Image upload, conversion, scaling and copy example. You have to set the file path for the uploaded file, and allowed file types for image uploads in the parameter file: 0014_img_upload_scale_param.cpy.

After upload the imgscale.c module will be used. With this module you can convert, scale and copy the uploaded image. The new image will be shown on the response page.

```
<body>
  <?cob PERFORM GET-VALUES
    PERFORM CREATE-OUT-NAME
    PERFORM IMGSCALE-AND-COPY
  ?>
  <h2>Image upload and scale</h2>
  <p>
    Image description: <?cob display trim(htm-filedesc) end-display ?><br>
    Image upload: <?cob display trim(htm-upload) end-display ?><br>
    Scaled Image: <?cob display trim(WS-OUT-NAME) end-display ?><br>
    <?cob display "<img src='../img/'trim(WS-OUT-NAME)'">" end-display ?><br>
  </p>
</body>
```

Use of imgscale.c module

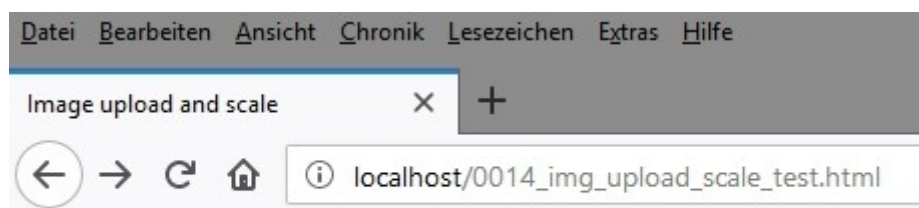


Image upload and scale

Image description:

PNG upload (max. 750 Kb): cyborg.png

0014_img_upload_scale_test

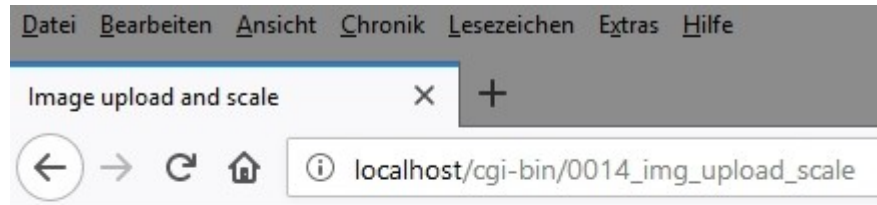


Image upload and scale

Image description: Cyborg test

Image upload: cyborg.png

Scaled Image: cyborg.jpg



0014_img_upload_scale

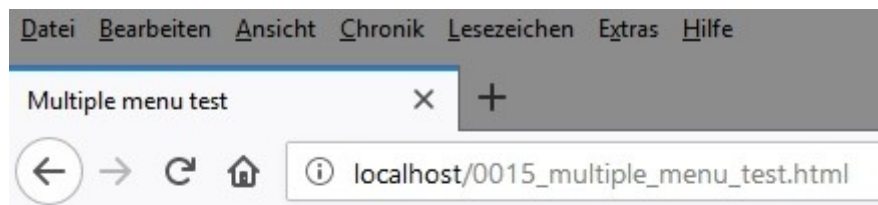
3.1.15 0015_multiple_menu

Multiple selection menu example. It is also an example of using the module "HTM2COB-POST-MULTI".

```
<?cob
*> use ; as a separator char
    call "HTM2COB-POST-MULTI" using "pizza", ";", ws-pizza end-call
    call "HTM2COB-SPEC-CHARS" using ws-pizza, htm-pizza end-call
?>

<p>
Your selected pizzas: <?cob display trim(htm-pizza) end-display ?><br>
(separator char: ";" )<br>
</p>
```

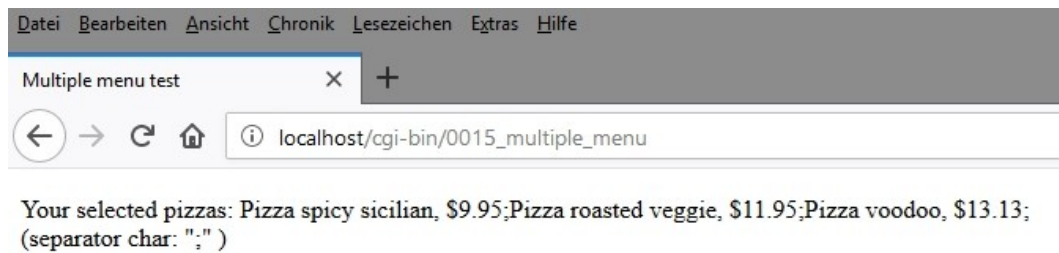
HTML with embedded COBOL



Multiple menu test

 A screenshot of a web form. It features a dropdown menu with four visible options: 'Pizza spicy sicilian', 'Pizza classic veggie', 'Pizza roasted veggie' (which is highlighted in blue), and 'Pizza deluxe'. Below the menu are two buttons: 'Send' and 'Zurücksetzen'.

0015_multiple_menu_test



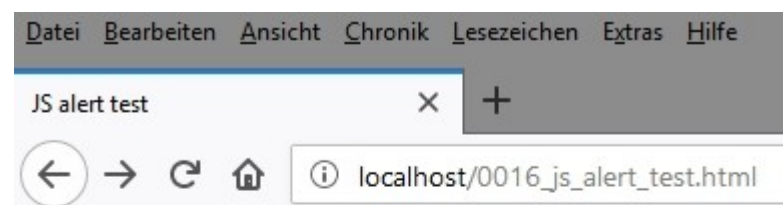
0015_multiple_menu

3.1.16 0016_js_alert

An example of Javascript: using onSubmit to validate inputted text.

```
<h2>JS alert test</h2>
<p>
<form name="formname" onSubmit="return length_check();" action="/cgi-bin/0016_
  <table border="0" cellpadding="0" cellspacing="4">
    <tr>
      <td align="right">First name (2-20 chars):</td>
      <td><input name="firstname" type="text" size="20" maxlength="20"></td>
    </tr>
```

HTML form with onSubmit

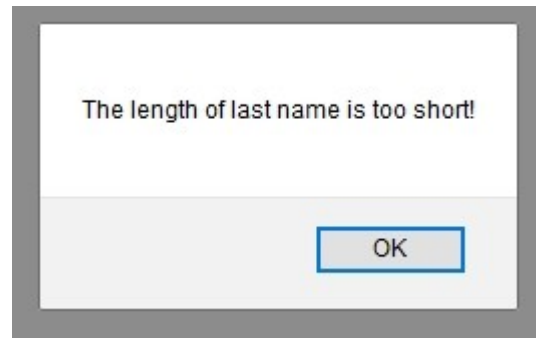


JS alert test

First name (2-20 chars):

Last name (2-20 chars):

0016_js_alert_test



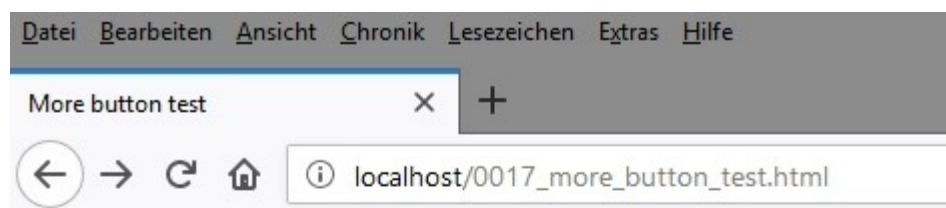
0016_js_alert_test_message

3.1.17 0017_more_button

This is an example for more buttons in a form.

```
<tr>
  <td align="left">Editor</td>
  <td><input name="pw_editor" type="password" size="20" maxlength="20"></td>
  <td><input type="button" value="Login" onClick="user_login('Editor')"></td>
</tr>
<tr>
  <td align="left">Admin</td>
  <td><input name="pw_admin" type="password" size="20" maxlength="20"></td>
  <td><input type="button" value="Login" onClick="user_login('Admin')"></td>
</tr>
<tr>
  <td align="left">SuperAdmin</td>
  <td><input name="pw_super_admin" type="password" size="20" maxlength="20"></td>
  <td><input type="button" value="Login" onClick="user_login('SuperAdmin')"></td>
</tr>
```

onClick with user_login() java script function

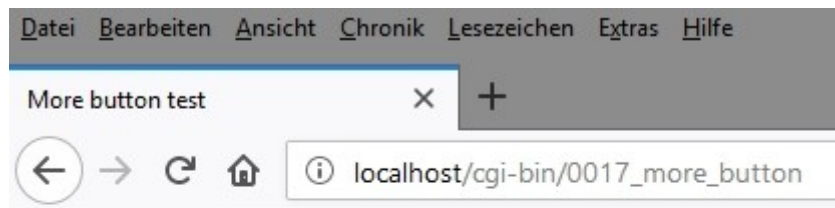


More button test

Name	Password	Registered users
Editor	<input type="password"/>	<input type="button" value="Login"/>
Admin	••••••	<input type="button" value="Login"/>
SuperAdmin	<input type="password"/>	<input type="button" value="Login"/>

(Passwords: Editor/qwerty1, Admin/qwerty2, SuperAdmin/qwerty3)

0017_more_button_test



More button test

Display information for Admin...

Display information for Guest...

0017_more_button

3.1.18 0018_link_submit

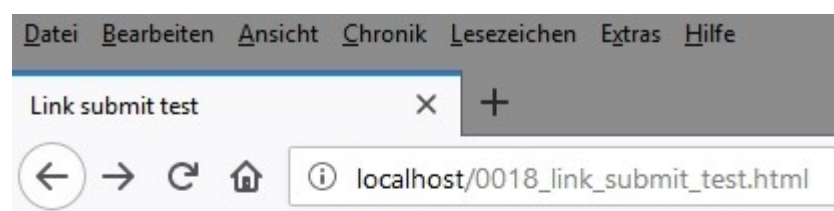
This is an example for a link submit.

```

</form>
</p>
<p>
<a href="javascript:document.hello_form.submit();">Click for hello!</a>
</p>

```

Hyperlink submit



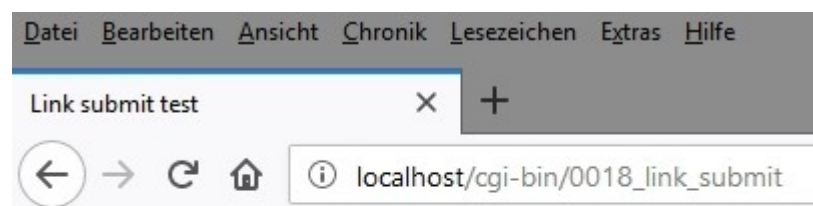
Link submit test

First name:

Last name:

[Click for hello!](#)

0018_link_submit_test



Link submit test

Hello László Erdős!

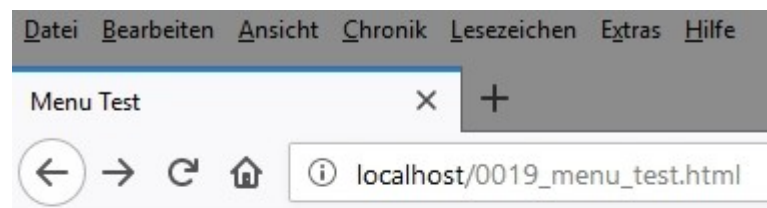
0018_link_submit

3.1.19 0019_menu

This is an example for an `` `` CSS menu.

```
<ul id="menu">
  <li><a href="/cgi-bin/0019_menu_1">Menu1</a></li>
  <li><a href="/cgi-bin/0019_menu_2">Menu2</a>
    <ul>
      <li><a href="/cgi-bin/0019_submenu_21">SubMenu21</a></li>
      <li><a href="/cgi-bin/0019_submenu_22">SubMenu22</a></li>
      <li><a href="/cgi-bin/0019_submenu_23">SubMenu23</a></li>
    </ul>
  </li>
  <li><a href="/cgi-bin/0019_menu_3">Menu3</a>
</ul>
```

`` `` menu definition

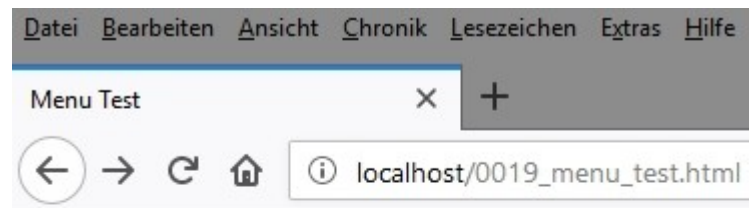


Menu Test



Menu Test page...

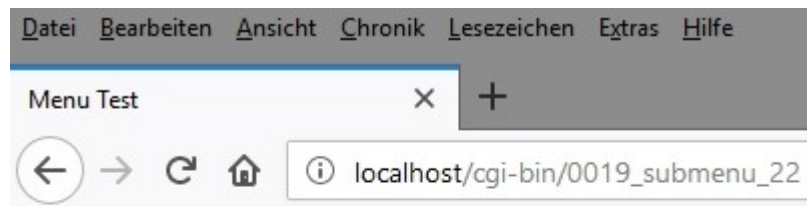
0019_menu_test



Menu Test



0019_menu_test_select



Menu Test

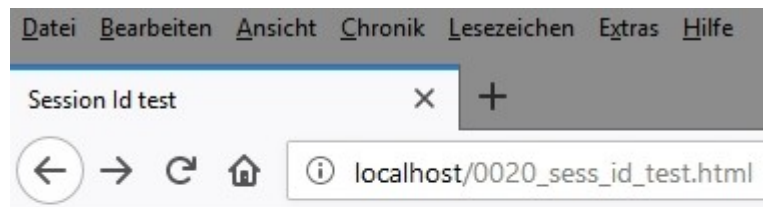


Sub-Menu22 page...

0019_submenu_22

3.1.20 0020_sess_id

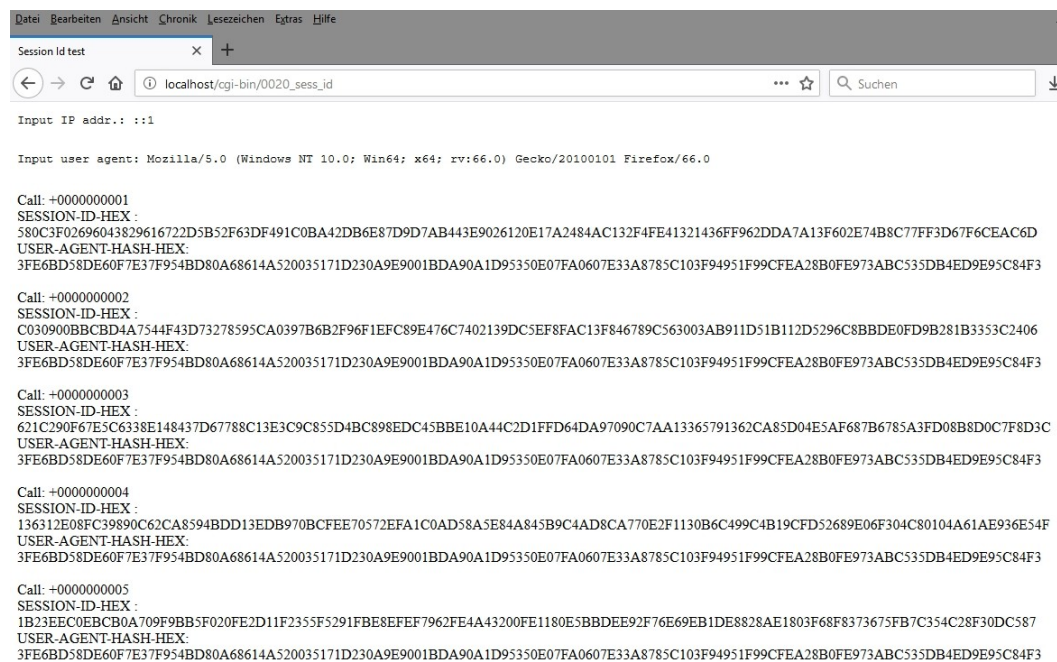
This example uses the "HTM2COB-SESSION-ID" module to generate 5 session-ids.



Session Id test

Send

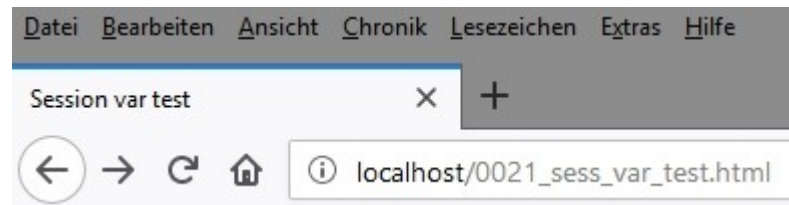
0020_sess_id_test



0020_sess_id

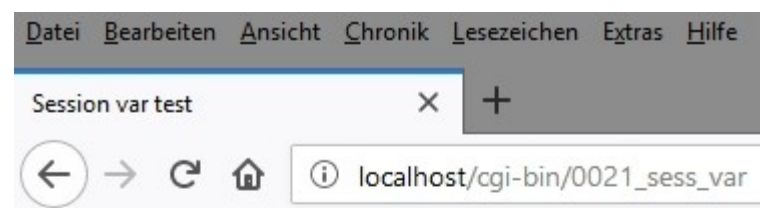
3.1.21 0021_sess_var

This example uses some session relevant sections: "HTM2COB-SESS-START", "HTM2COB-SESS-SET", "HTM2COB-SESS-GET" and "HTM2COB-SESS-DEL". Session data will be written in the session files: "/tmp/0021_session.dat" and "/tmp/0021_session_var.dat".



Session var test

0021_sess_var_test



Step 1: start session

Step 2: set session var

HTM2COB-SESS-VAR-NAME: mydog
HTM2COB-SESS-VAR-VALUE: Morgo

Step 3: get session var

HTM2COB-SESS-VAR-NAME: mydog
HTM2COB-SESS-VAR-VALUE: Morgo

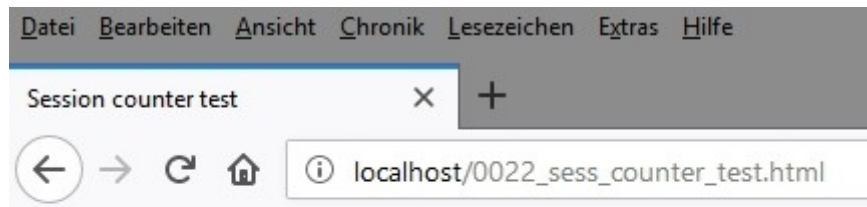
Step 4: del session var

HTM2COB-SESS-VAR-NAME: mydog

0021_sess_var

3.1.22 0022_sess_counter

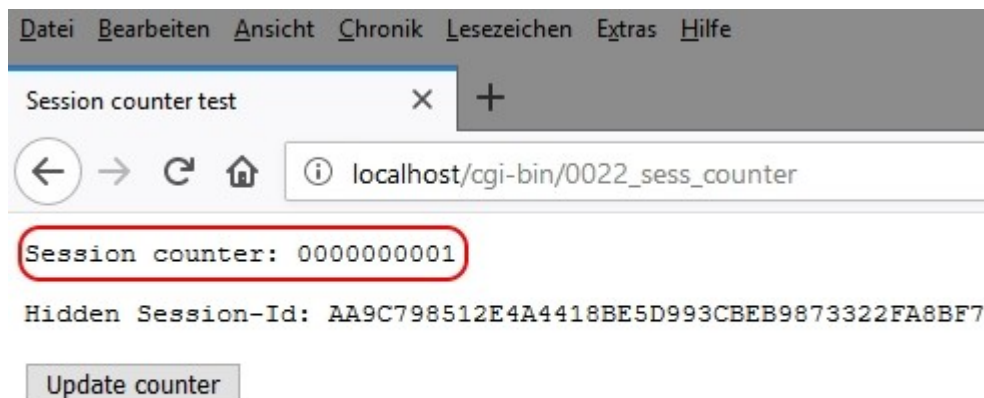
This is an example for a counter. The counter value will be saved in the session file, and it will be updated after every call.



Session counter test

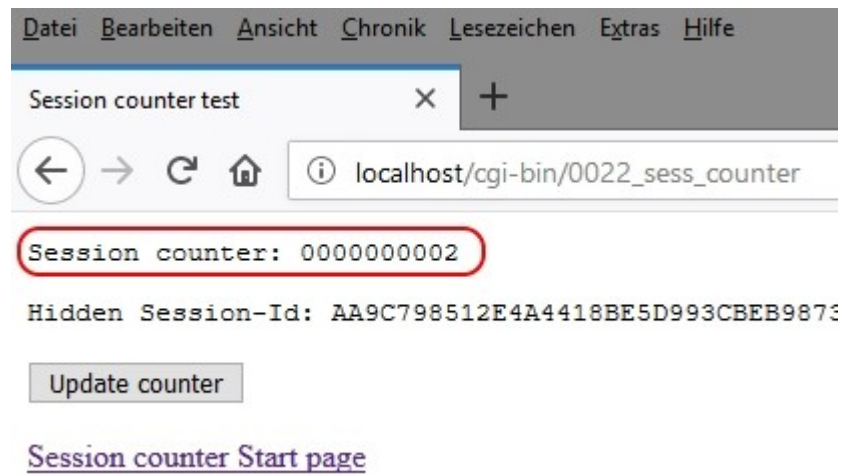
Send

0022_sess_counter_test

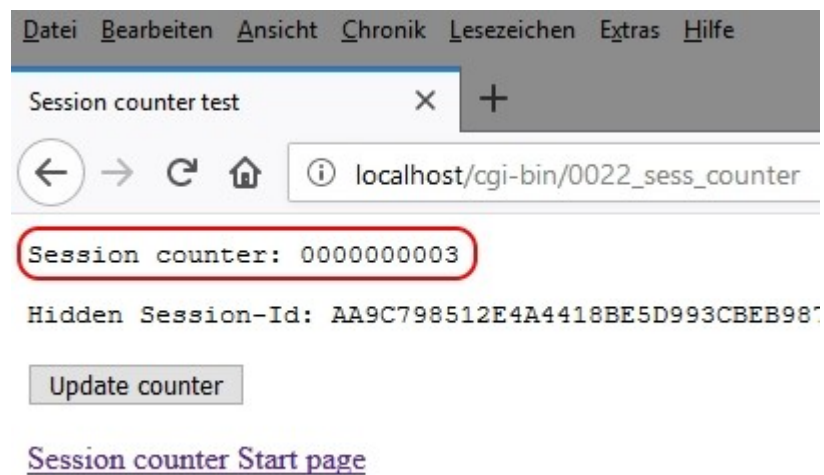


[Session counter Start page](#)

0022_sess_counter_1



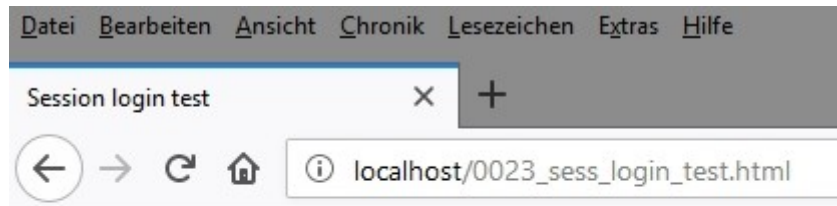
0022_sess_counter_2



0022_sess_counter_3

3.1.23 0023_sess_login

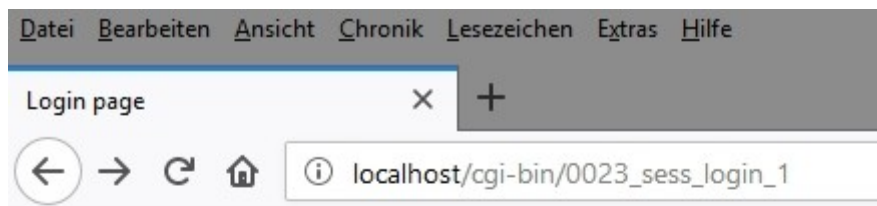
This is a simple example for session tracking with login. The user name will be saved in the session file after successful login. The existence of this user name will be checked on the following pages. If the user visits the login page later, then a new session-id will be created.



Session login test

Send

0023_sess_login_test



HTM2COB-HIDDEN-SESSION-ID: DF7258233D32CFAE7F392EAABE4

Login page

Name:

Password:

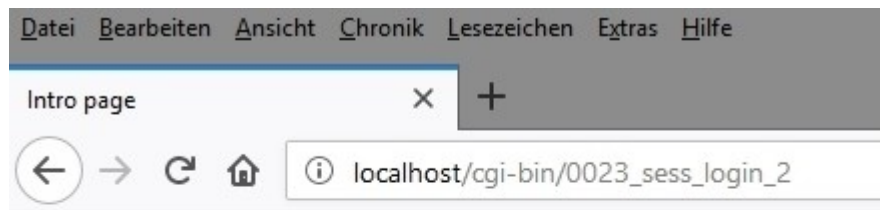
Valid names and passwords: "admin/qwerty" or "user1/pw123".

Submit

Zurücksetzen

[Session login start page](#)

0023_sess_login_1

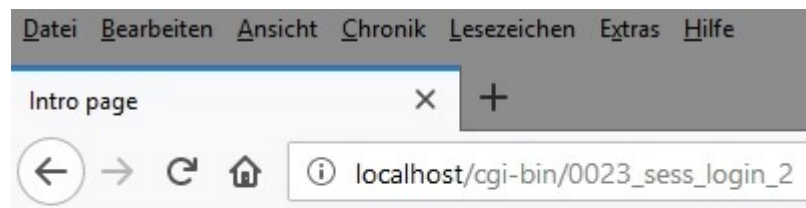


HTM2COB-HIDDEN-SESSION-ID: F033578954B9AC441BFC9EC6B7E

No access

Login page

0023_sess_login_2_unsuccessful



HTM2COB-HIDDEN-SESSION-ID: 8A643FA31EFFBE301CECA95

Intro page

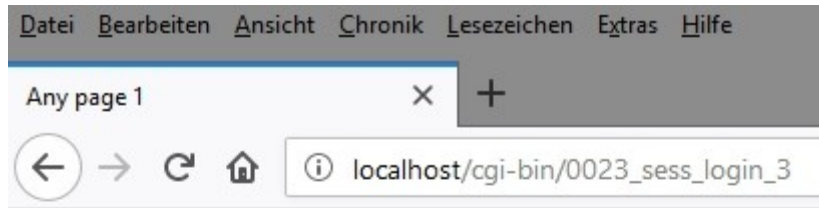
Hello user1

Any page 1

Logoff

[Session login start page](#)

0023_sess_login_2



HTM2COB-HIDDEN-SESSION-ID: 8A643FA31EFFBE301CECA99

Any page 1

Hello user1

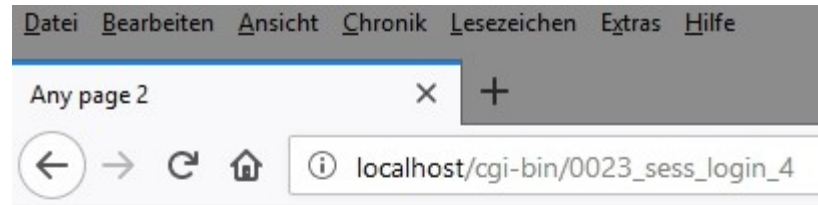
Any page 2

Intro page

Logoff

[Session login start page](#)

0023_sess_login_3



HTM2COB-HIDDEN-SESSION-ID: 8A643FA31EFFBE301CECA99

Any page 2

Hello user1

Any page 1

Intro page

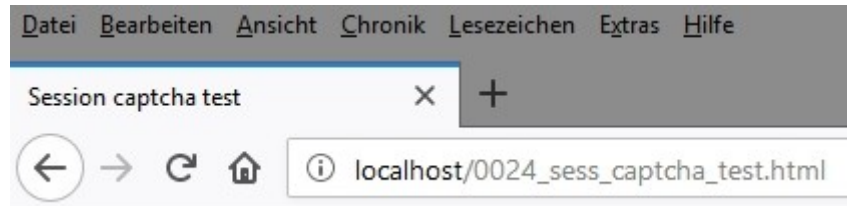
Logoff

[Session login start page](#)

0023_sess_login_4

3.1.24 0024_sess_captcha

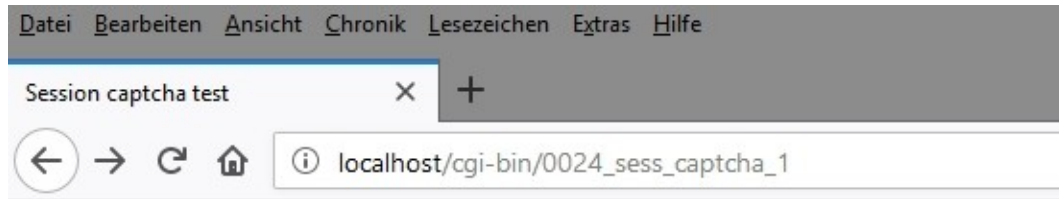
This is a captcha example. The text of the image captcha will be saved in the session. This text will be checked at submit. It is also an example of using the module "imgcaptcha".



Session captcha test

Send

0024_sess_captcha_test



Write your comment

Your name:

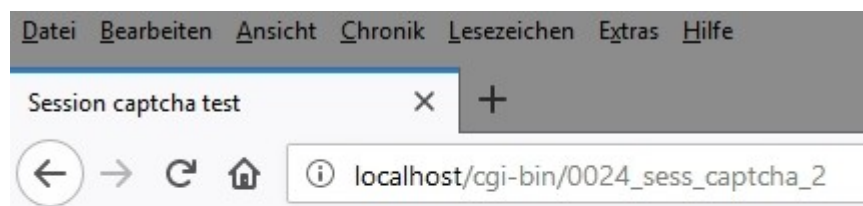
Comment:



CAPTCHA number:

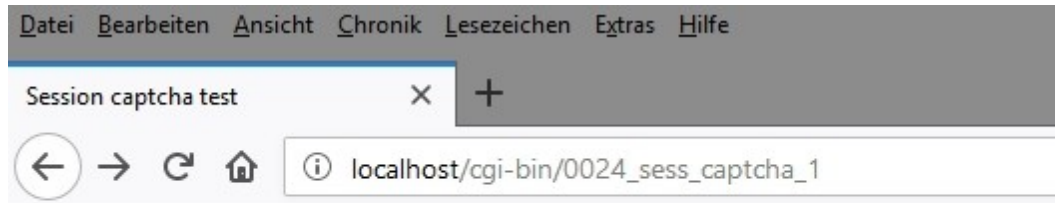
[Session captcha test start page](#)

0024_sess_captcha_1_wrong



Captcha verification failed, please try again.

0024_sess_captcha_2_unsuccessful



Write your comment

Your name:

Comment:

Real programmers don't write specs. Users should consider themselves lucky to get any programs at all and take what they get.

Real programmers don't comment their code. If it was hard to write, it should be hard to read.

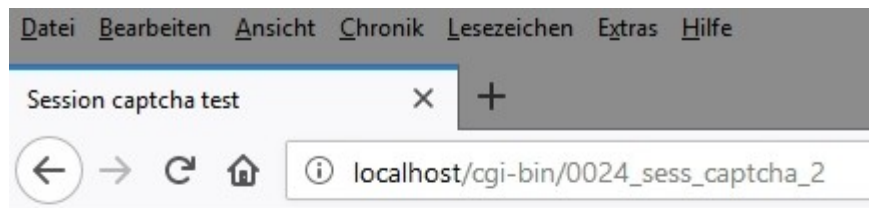
Real programmers never work 9 to 5. If any real programmers are around at 9 am, it's because they were up all night.



CAPTCHA number:

[Session captcha test start page](#)

0024_sess_captcha_1



Your comment was accepted!

Hello Hit Girl

Your comment: Real programmers don't write specs. Users should
was hard to write, it should be hard to read. Real programmers nev

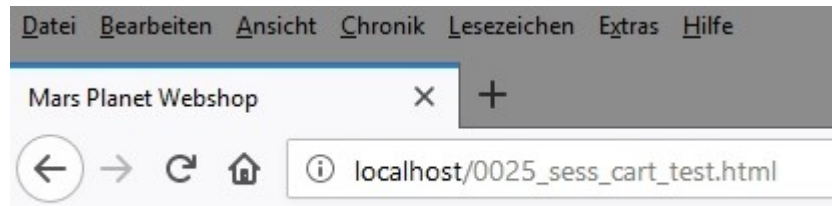
Session captcha test page

[Session captcha test start page](#)

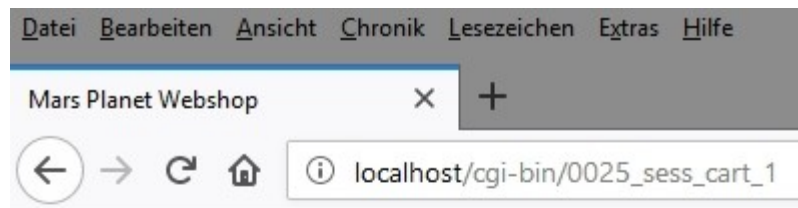
0024_sess_captcha_2

3.1.25 0025_sess_cart

This is a shopping cart example. You can add or delete items from several categories to the shopping cart. These things are stored in the session file. At the end you can do a checkout and pay.



0025_sess_cart_test



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB4C

Mars Planet Webshop

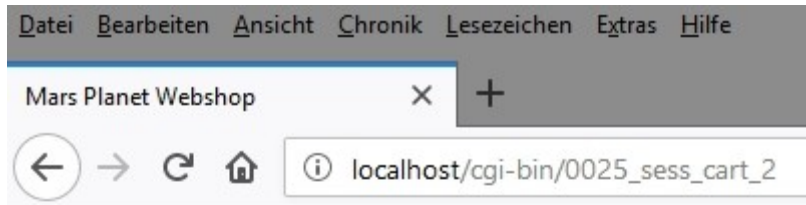
Sorry, there is no return ticket back to Earth...

Choose a category

Nr.	Category	Select Category
1	Vital items	Go to Category...
2	Robots and Mars-Rovers	Go to Category...
3	Real estate	Go to Category...
4	Travel	Go to Category...

[Mars Planet Webshop start page](#)

0025_sess_cart_1



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB407

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Enter quantities for items

Nr.	Item	Price (\$)	Quantity
1	Oxygen (1 liter)	0.01	<input type="text" value="0100"/>
2	Water (1 liter)	0.50	<input type="text" value="0002"/>
3	Potato (1 Kg)	2.50	<input type="text" value="0001"/>
4	Mars beer (0,33 liter)	3.99	<input type="text" value="0003"/>
5	Mars VitalMix (0,33 liter)	9.99	<input type="text" value="0004"/>

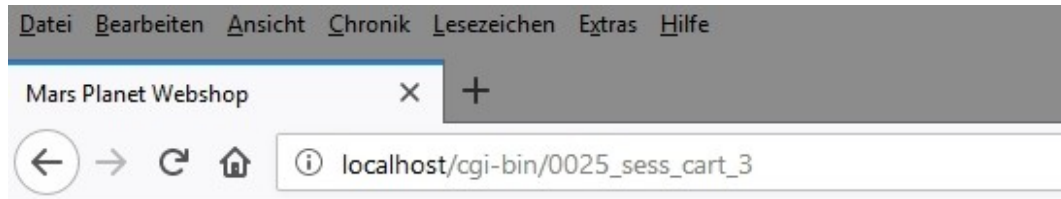
Add to Cart

Zurücksetzen

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0025_sess_cart_2_select_1



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB407A6271D1C034BA89E6

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Cart Contents

Item-Nr.	Item	Price (\$)	Quantity	Subtotal Price (\$)
Cat1-Item1	Oxygen (1 liter)	0.01	100	1.00
Cat1-Item2	Water (1 liter)	0.50	2	1.00
Cat1-Item3	Potato (1 Kg)	2.50	1	2.50
Cat1-Item4	Mars beer (0,33 liter)	3.99	3	11.97
Cat1-Item5	Mars VitalMix (0,33 liter)	9.99	4	39.96

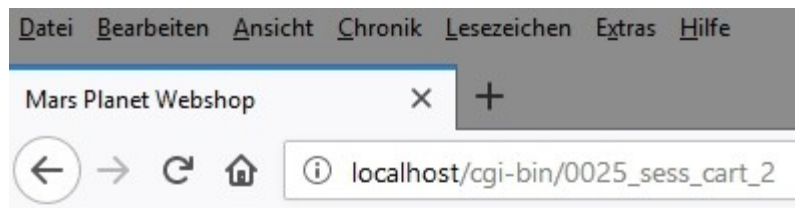
Total Price (\$): 56.43

Checkout

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0025_sess_cart_3_selected_1



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB40

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Enter quantities for items

Nr.	Item	Price (\$)	Quantity
1	Universal Mars Robot	600.00	<input type="text" value="0001"/>
2	Cherry 2000 Female Android	2,000.00	<input type="text" value="0001"/>
3	Mars Rover Speed	15,000.00	<input type="text" value="0001"/>
4	Mars Rover Off Road 4x4	25,000.00	<input type="text" value="0000"/>
5	Mars Flying Saucer	80,000.00	<input type="text" value="0000"/>

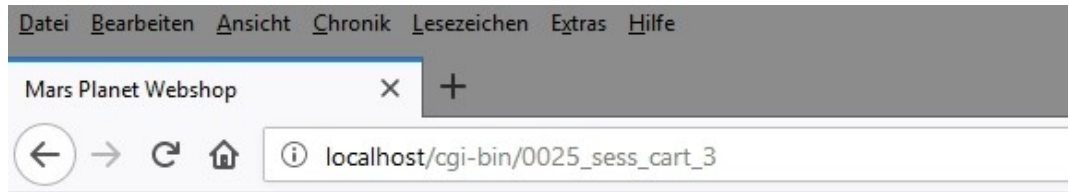
Add to Cart

Zurücksetzen

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0025_sess_cart_2_select_2



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB407A6271D1C034BA89E68CFA

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Cart Contents

Item-Nr.	Item	Price (\$)	Quantity	Subtotal Price (\$)
Cat1-Item1	Oxygen (1 liter)	0.01	100	1.00
Cat1-Item2	Water (1 liter)	0.50	2	1.00
Cat1-Item3	Potato (1 Kg)	2.50	1	2.50
Cat1-Item4	Mars beer (0,33 liter)	3.99	3	11.97
Cat1-Item5	Mars VitalMix (0,33 liter)	9.99	4	39.96
Cat2-Item1	Universal Mars Robot	600.00	1	600.00
Cat2-Item2	Cherry 2000 Female Android	2,000.00	1	2,000.00
Cat2-Item3	Mars Rover Speed	15,000.00	1	15,000.00

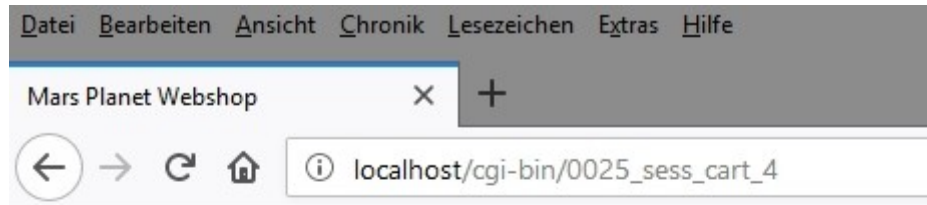
Total Price (\$): 17,656.43

Checkout

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0025_sess_cart_3_selected_2



HTM2COB-HIDDEN-SESSION-ID: 7E9D289CDF83A30E9CFB407A6271D1

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Checkout

Total Price (\$): 17,656.43

Please transfer the total amount to the given bank account. **IBAN: ...**

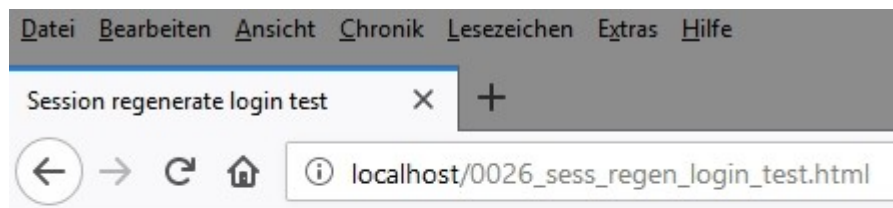
[Mars Planet Webshop start page](#)

0025_sess_cart_4

3.1.26 0026_sess_regen_login

This example is similar to the example 0023_sess_login. But here the session-id is regenerated after every request. It is also an example for the usage of section HTM2COB-SESS-REGENERATE.

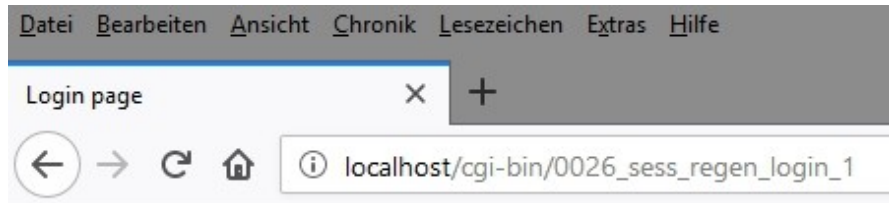
The user name will be saved in the session file after successful login. The existence of this user name will be checked on the following pages. If the user visits the login page later, then a new session-id will be created.



Session regenerate login test

Send

0026_sess_regen_login_test



HTM2COB-HIDDEN-SESSION-ID: D8173BAF4159EF5493F105D5B54B7

Login page

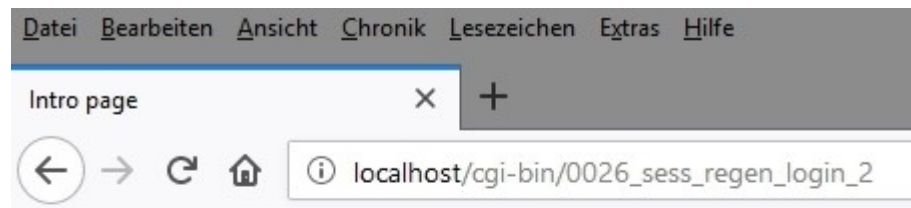
Name:

Password:

Valid names and passwords: "admin/qwerty" or "user1/pw123".

[Session regenerate login start page](#)

0026_sess_regen_login_1



HTM2COB-HIDDEN-SESSION-ID: 1614A82D6DB4254244A97CB3CFC2A

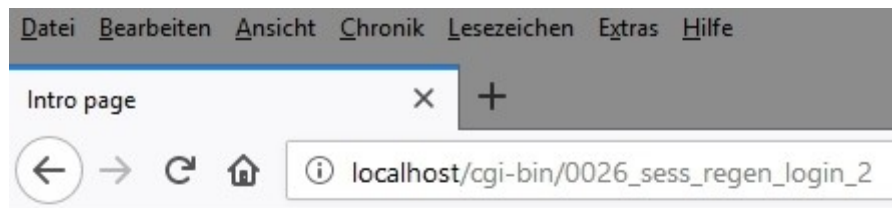
after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 428595C81FDEBD5A3BB348566905I

No access

Login page

0026_sess_regen_login_2_unsuccessful



HTM2COB-HIDDEN-SESSION-ID: 10B0639334140B58C645CB7C8FA7

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 252923B491CA0246DEC27DCB7D20

Intro page

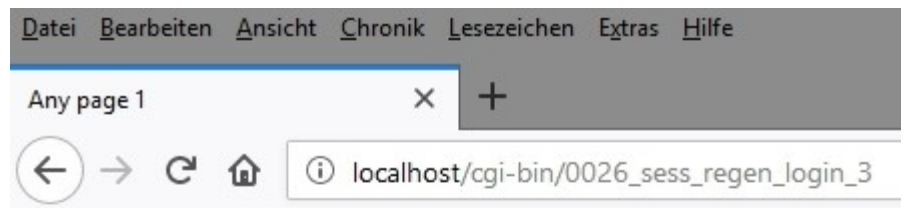
Hello admin

Any page 1

Logoff

[Session regenerate login start page](#)

0026_sess_regen_login_2



HTM2COB-HIDDEN-SESSION-ID: 252923B491CA0246DEC27DCB7D2C

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: CEBCEC29916DF38A8F4CACE5C9E4

Any page 1

Hello admin

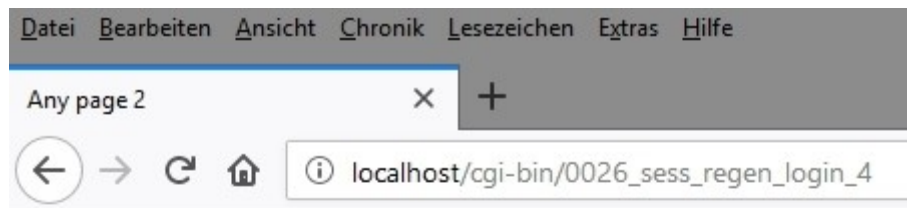
Any page 2

Intro page

Logoff

[Session regenerate login start page](#)

0026_sess_regen_login_3



HTM2COB-HIDDEN-SESSION-ID: CEBCEC29916DF38A8F4CACE5C9E4

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 3E352BDB3E18001E681ADF84A95B

Any page 2

Hello admin

Any page 1

Intro page

Logoff

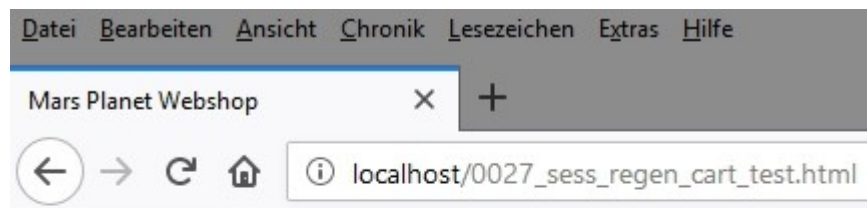
[Session regenerate login start page](#)

0026_sess_regen_login_4

3.1.27 0027_sess_regen_cart

This example is similar to the example 0025_sess_cart. But here the session-id is regenerated after every request. It is also an example for the usage of section HTM2COB-SESS-REGENERATE.

This is a shopping cart example. You can add or delete items from several categories to the shopping cart. These things are stored in the session file. At the end you can do a checkout and pay.

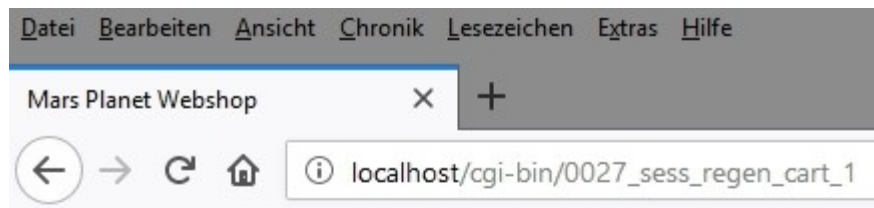


Welcome to the Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Start

0027_sess_regen_cart_test



HTM2COB-HIDDEN-SESSION-ID: 482B33EC555F9E463372262F012

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: ABC0C9482F8DA483D2493E03E3E

Mars Planet Webshop

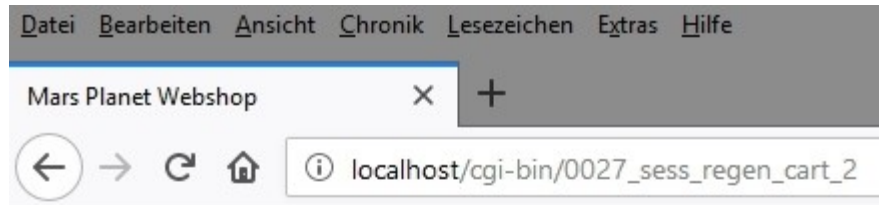
Sorry, there is no return ticket back to Earth...

Choose a category

Nr.	Category	Select Category
1	Vital items	Go to Category...
2	Robots and Mars-Rovers	Go to Category...
3	Real estate	Go to Category...
4	Travel	Go to Category...

[Mars Planet Webshop start page](#)

0027_sess_regen_cart_1



HTM2COB-HIDDEN-SESSION-ID: ABC0C9482F8DA483D2493E03E3B

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: B3D3909820EBDA6D3697071EF14

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Enter quantities for items

Nr.	Item	Price (\$)	Quantity
1	Oxygen (1 liter)	0.01	<input type="text" value="0020"/>
2	Water (1 liter)	0.50	<input type="text" value="0003"/>
3	Potato (1 Kg)	2.50	<input type="text" value="0001"/>
4	Mars beer (0,33 liter)	3.99	<input type="text" value="0002"/>
5	Mars VitalMix (0,33 liter)	9.99	<input type="text" value="0004"/>

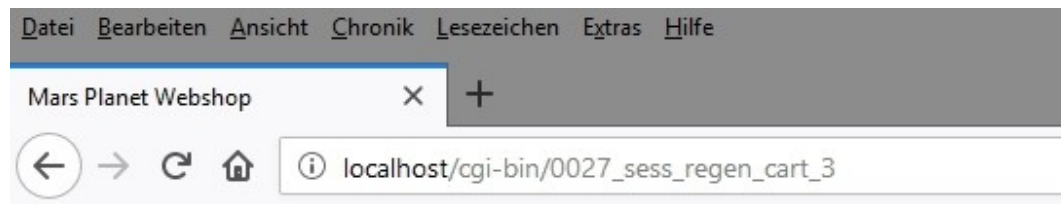
Add to Cart

Zurücksetzen

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0027_sess_regen_cart_2_select_1



HTM2COB-HIDDEN-SESSION-ID: B3D3909820EBDA6D3697071EF1433B5C2A4EF404

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 8A86298B65DFAF809A1DEE1E987653E3AFCF84E2

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Cart Contents

Item-Nr.	Item	Price (\$)	Quantity	Subtotal Price (\$)
Cat1-Item1	Oxygen (1 liter)	0.01	20	0.20
Cat1-Item2	Water (1 liter)	0.50	3	1.50
Cat1-Item3	Potato (1 Kg)	2.50	1	2.50
Cat1-Item4	Mars beer (0,33 liter)	3.99	2	7.98
Cat1-Item5	Mars VitalMix (0,33 liter)	9.99	4	39.96

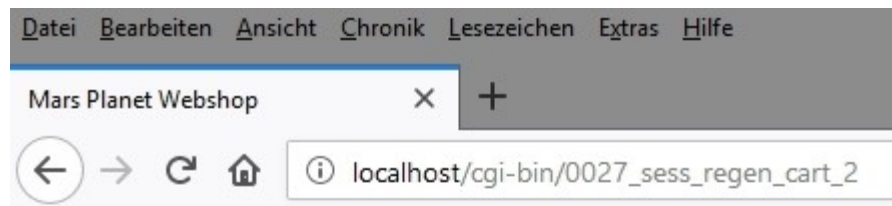
Total Price (\$): 52.14

Checkout

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0027_sess_regen_cart_3_selected_1



HTM2COB-HIDDEN-SESSION-ID: 69F150AC40831A39252A95381130

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 01B84414E7BE8E2F16E4ED26E130

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Enter quantities for items

Nr.	Item	Price (\$)	Quantity
1	Universal Mars Robot	600.00	<input type="text" value="0001"/>
2	Cherry 2000 Female Android	2,000.00	<input type="text" value="0001"/>
3	Mars Rover Speed	15,000.00	<input type="text" value="0000"/>
4	Mars Rover Off Road 4x4	25,000.00	<input type="text" value="0000"/>
5	Mars Flying Saucer	80,000.00	<input type="text" value="0001"/>

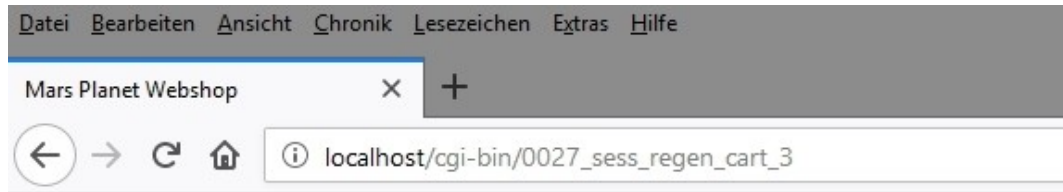
Add to Cart

Zurücksetzen

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0027_sess_regen_cart_2_select_2



HTM2COB-HIDDEN-SESSION-ID: 669823D4AD72EEB9BF739F5F5535E9EC2D87A43E6106

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: 8E5960337D6306ADCC5EF0DF2F9E58791C6B6BD48A03

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Cart Contents

Item-Nr.	Item	Price (\$)	Quantity	Subtotal Price (\$)
Cat1-Item1	Oxygen (1 liter)	0.01	20	0.20
Cat1-Item2	Water (1 liter)	0.50	3	1.50
Cat1-Item3	Potato (1 Kg)	2.50	1	2.50
Cat1-Item4	Mars beer (0,33 liter)	3.99	2	7.98
Cat1-Item5	Mars VitalMix (0,33 liter)	9.99	4	39.96
Cat2-Item1	Universal Mars Robot	600.00	1	600.00
Cat2-Item2	Cherry 2000 Female Android	2,000.00	1	2,000.00
Cat2-Item5	Mars Flying Saucer	80,000.00	1	80,000.00

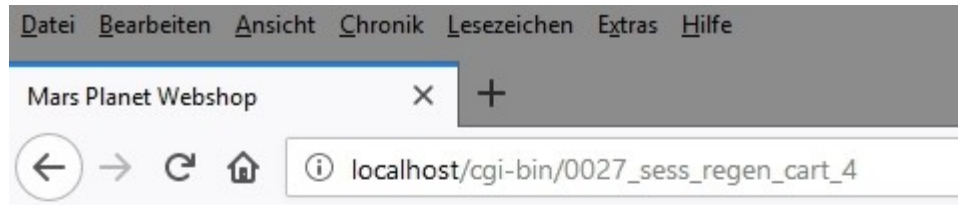
Total Price (\$): 82,652.14

Checkout

Mars Planet Webshop Categories

[Mars Planet Webshop start page](#)

0027_sess_regen_cart_3_selected_2



HTM2COB-HIDDEN-SESSION-ID: 8E5960337D6306ADCC5EF0DF2F9E5879

after regenerate session

HTM2COB-HIDDEN-SESSION-ID: F5029519878E0460737E4A09F80E3D19

Mars Planet Webshop

Sorry, there is no return ticket back to Earth...

Checkout

Total Price (\$): 82,652.14

Please transfer the total amount to the given bank account. **IBAN:** ...

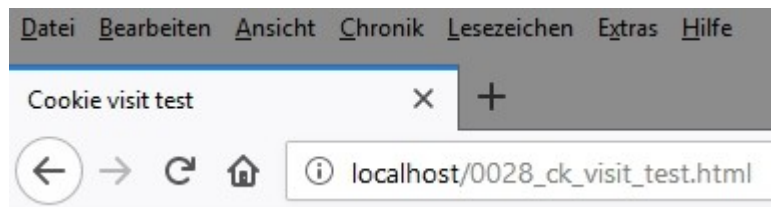
[Mars Planet Webshop start page](#)

0027_sess_regen_cart_4

3.1.28 0028_ck_visit

This is a cookie visitor test. The cookie does not exist at first try, and after the user has visited the page it will be set. At the second try the cookie will be successfully read, and the visitor be recognized. For cookies you have to set some parameters in the parameter file 0028_ck_visit_param.cpy, and in the HTML file you have to implement the section HTM2COB-SET-COOKIE.

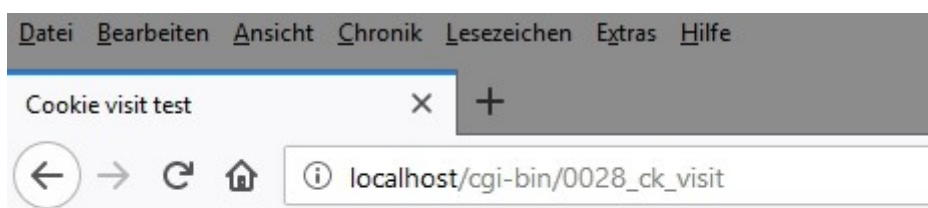
```
*>-----
*> Max length of the HTM2COB-HTTP-COOKIE field. This has the value from
*> the HTTP_COOKIE environment variable.
78 HTM2COB-HTTP-COOKIE-MAX-LEN      VALUE 10000.
*> If it set to "YES", then the section HTM2COB-SET-COOKIE will be used. That
*> means, you have to implement this section in your HTML file.
>>DEFINE USE-HTM2COB-SET-COOKIE-SECTION AS "YES"
```



Cookie visit test

Send

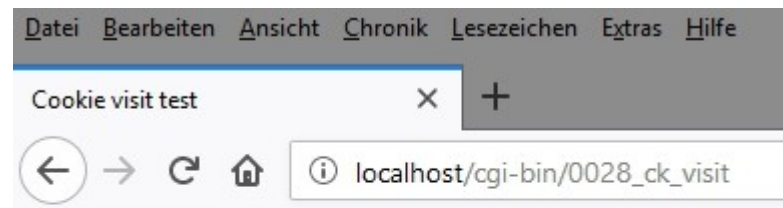
0028_ck_visit_test



Cookie visit test

You have not been here yet, or you can not save cookies. Try again!

0028_ck_visit_try_1



Cookie visit test

You have been here before.

0028_ck_visit_try_2

3.1.29 0029_ck_save_data

TODO

3.1.30 0030_ck_param

TODO

3.1.31 0031_ck_sess_login

TODO

3.1.32 0032_ck_sess_captcha

TODO

3.1.33 0033_ck_sess_cart

TODO

3.1.34 0034_ck_sess_regen_login

TODO

3.1.35 0035_ck_sess_regen_cart

TODO

3.1.36 0036_ajax_hello

TODO

3.1.37 0037_ajax_query_str

TODO

3.1.38 0038_jq_form_validate

TODO

3.1.39 0039_jq_jcarousel

TODO

3.2 Projects

TO DO

4. Bibliography and references

1. Undine Schrader: Webprogrammierung mit Perl und CGI, m. CD-ROM
ISBN: 978-3-772-36880-6, Franzis (2004)
2. Thomas Theis: Einstieg in PHP 5.4 und MySQL 5.5
ISBN 978-3-836-21877-1, Galileo Press, Bonn 2012
3. Lynn Beighley: jQuery für Dummies
ISBN: 978-3-527-70863-5, WILEY-VCH Verlag GmbH & Co. KgaA,
Weinheim 2012
4. Kyle Honeycutt: Building Bitcoin Websites
ISBN: 978-1-534-94544-9, 2016
5. Mario Heiderich, Christian Matthies, fukami, Johannes Dahse:
Sichere Webanwendungen
ISBN: 978-3-836-21194-9, Galileo Press, Bonn 2009
6. Michael Cross:
Developer's Guide to Web Application Security
ISBN: 978-1-597-49061-0, Syngress Publishing, Inc.