
INSTALLING THE XEMISTRY WEB SKETCHER ON LINUX/BSD/UNIX SYSTEMS RUNNING THE APACHE2 WEB SERVER

This is a step-by-step introduction to set up the Web sketcher application on Unix-style systems. This manual assumes that you are familiar with basic system administration, and that you have sufficient permissions to install executables, administer Apache setting, and create new Websites.

STEP 1: VERIFY THAT YOU HAVE A WORKING WEB SERVER ON YOUR SYSTEM

The Apache web server may not be installed, or enabled, on all system installations. Make sure that your server has a properly configured and running Apache2 server, and that you can load pages and dynamically generated content from other sites, or just the default install demo pages, on the installation system using a standard browser such as Firefox.

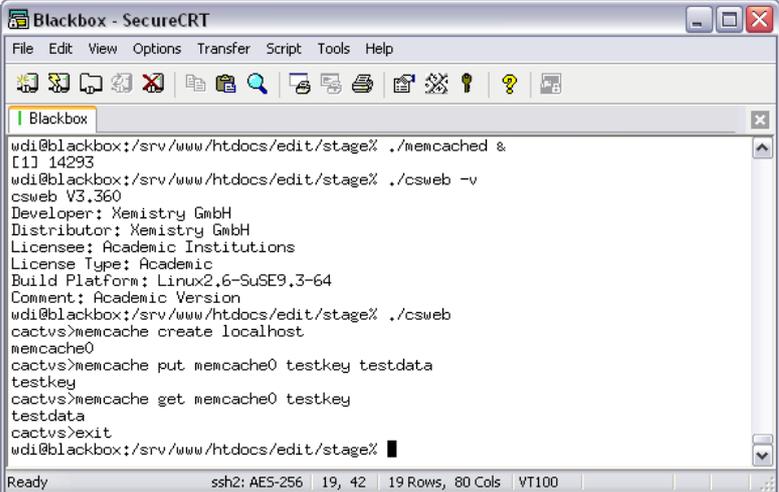
STEP 2: EXPAND THE SKETCHER SOURCE PACKAGE

Create an empty directory in the Web server file space. Traditionally, the application is installed in a root directory of the server such as */edit* or */sketcher* which may be located in the file system at */srv/www/htdocs/edit* or */srv/www/htdocs/sketcher*, respectively, but this is not a hard requirement.

Next, download the compressed tar-file of the package and expand it in the new Web server directory. You will see a single new directory called *stage*, which contains all software and templates needed for the sketcher application proper.

In the staging directory, you will find a generic interpreter executable named *csweb*. If you want to use a different interpreter in your set-up, for example for communication with a different state store service which is not supported in a generic interpreter, such as NCBI *queuman*, this is the time to replace the interpreter. You can either directly overwrite the default interpreter, or give it a different name and then later adjust the installation Makefile. In any case, the executable should be copied to the staging directory first.

It is a good idea to verify that the executables work on the selected installation platform. Here are a few test commands:



```
Blackbox - SecureCRT
File Edit View Options Transfer Script Tools Help
Blackbox
wdi@blackbox:/srv/www/htdocs/edit/stage% ./memcached &
[1] 14293
wdi@blackbox:/srv/www/htdocs/edit/stage% ./csweb -v
csweb V3.360
Developer: Xemistry GmbH
Distributor: Xemistry GmbH
Licensee: Academic Institutions
License Type: Academic
Build Platform: Linux2.6-SuSE9.3-64
Comment: Academic Version
wdi@blackbox:/srv/www/htdocs/edit/stage% ./csweb
cactvs>memcache create localhost
memcache0
cactvs>memcache put memcache0 testkey testdata
testkey
cactvs>memcache get memcache0 testkey
testdata
cactvs>exit
wdi@blackbox:/srv/www/htdocs/edit/stage% █
Ready      ssh2: AES-256 | 19, 42 | 19 Rows, 80 Cols | VT100
```

There should be no error messages, especially not about missing shared libraries. This should also be verified by running these commands as the Apache server user, using its real execution environment.

STEP 2: CHECK AND UPDATE THE WEB SERVER CONFIGURATION

In order to run the sketcher Web application, the Web server needs to support FCGI. Simple CGI is not fast enough for interactive response times. For Apache, FCGI comes in two flavors: Legacy FCGI and FCGID. Both will work, but server configuration is slightly different. For the details, study the module documentation. Not all Apache installations enable one of these modules by default. Verify with your server administrator that the Web server you are planning to use does support it.

In any case, there need to be two items present in the Apache server configuration files. Their exact location depends very much in the configuration of the system at hand, so a recursive grep in */etc/apache2* or whatever the place for the Apache configuration file is located is advised.

First, there needs to be an association between the file suffix *.fcgi* and the installed FCGI or FCGID handler. This entry is probably already present in the configuration files and looks something like

```
AddHandler fcgid-script .fcgi
```

The exact syntax depends on the used FCGI module. The next step is to configure the sketcher installation directory to support (F)CGI scripts run in that directory. Add a per-directory configuration section like

```
<Directory "/srv/www/htdocs/edit">
    AllowOverride None
    Options +ExecCGI
    Order allow,deny
    Allow from all
</Directory>
```

with the file system name of your chosen installation location. After typing this, restart the Web server, for example by running

```
/etc/init.d/apache2 restart
```

and verify that there are no error messages.

If you do not have sufficient permissions for these server configuration tasks, talk to your server administrator. The sketcher requires these configurations and is guaranteed to fail to work if these are not properly set up.

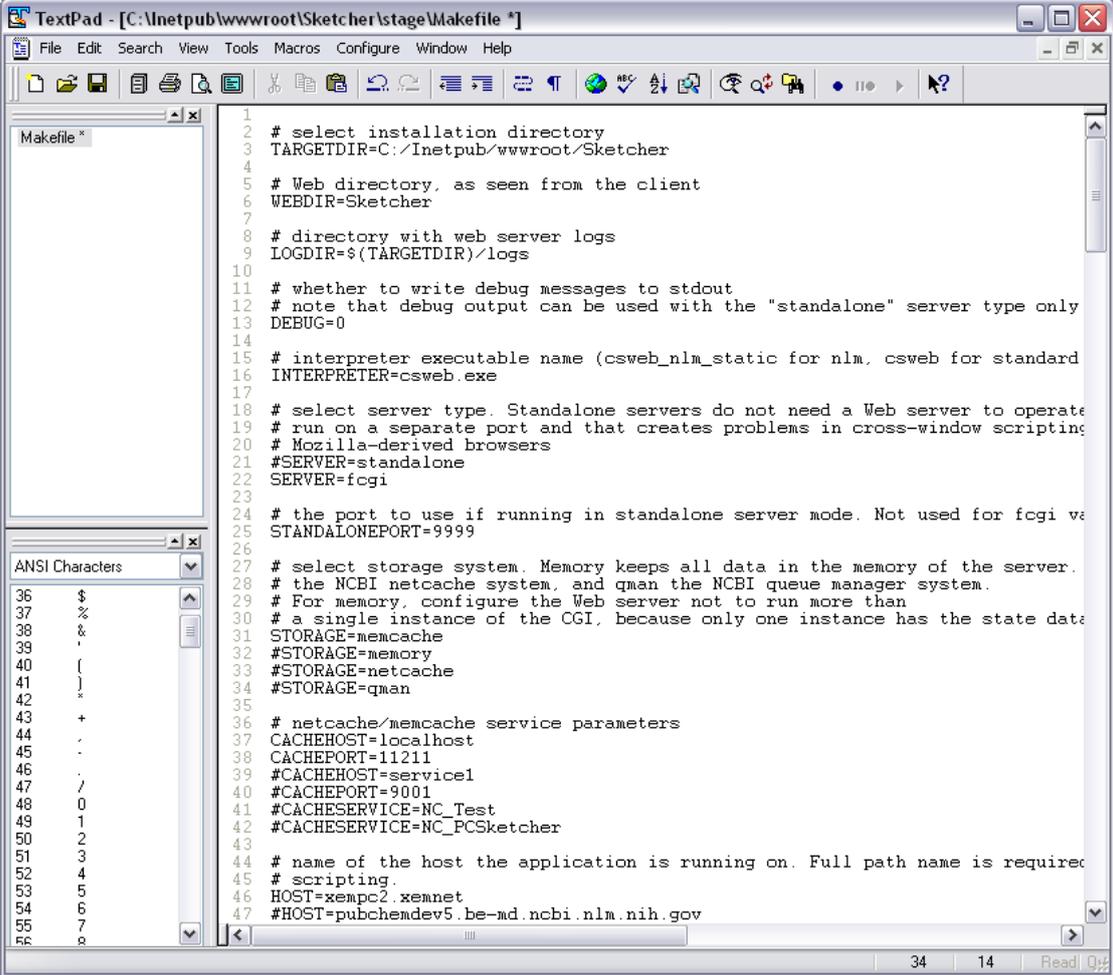
STEP 3: CONFIGURING THE SKETCHER INSTALLATION

In the staging directory, there is a file called Makefile. Open it in a plain text editor like *vi* or *keedit* and configure the Makefile variables in the first lines according to your set-up. The most important three items are:

TARGETDIR: The full path of the installation directory, directly below the stage directory. Use forward slashes, not backslashes for directory separation.

WEBDIR: The path to the installation directory from the Web root, as seen from a Web browser. Do not use a leading slash.

HOST: The full domain-resolved name of the server. The domain part is important if you want to support data exchange with Web forms which are hosted in the same fundamental domain, but not the same subdomain. If this is not configured correctly, data exchange via JavaScript functions may be seen as cross-site scripting security violation and suppressed by the Web browser, resulting in the inability to transfer data between a recipient application form and the sketcher.



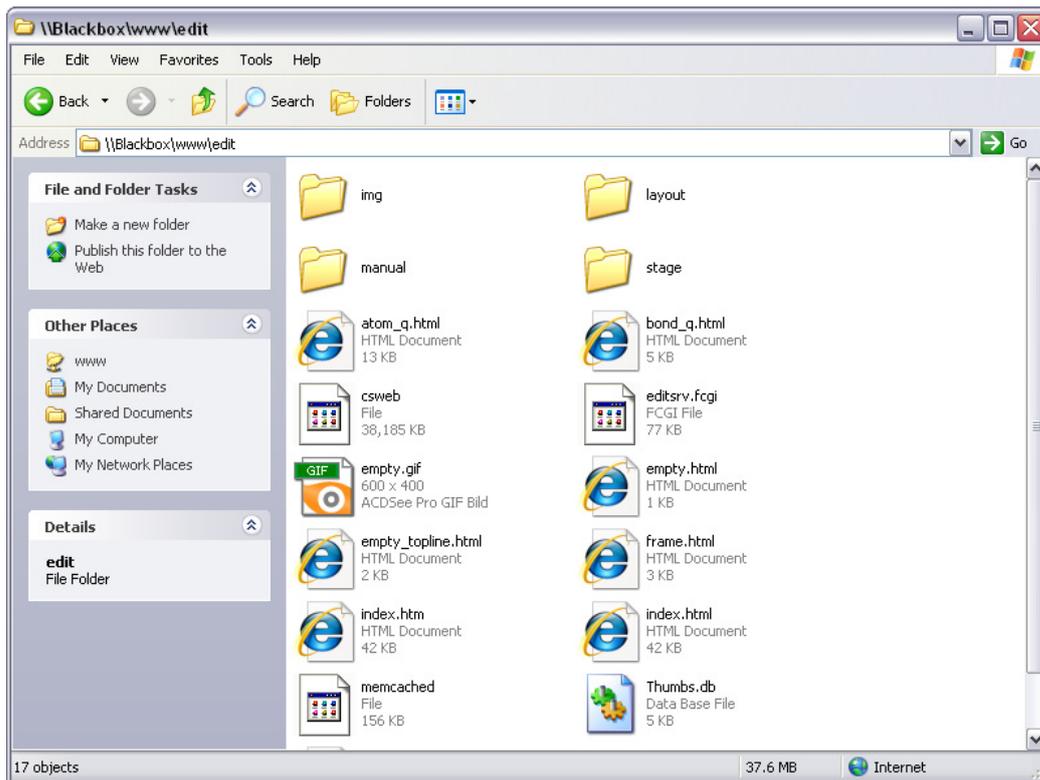
```
1 # select installation directory
2 TARGETDIR=C:/inetpub/wwwroot/Sketcher
3
4
5 # Web directory, as seen from the client
6 WEBDIR=Sketcher
7
8 # directory with web server logs
9 LOGDIR=$(TARGETDIR)/logs
10
11 # whether to write debug messages to stdout
12 # note that debug output can be used with the "standalone" server type only
13 DEBUG=0
14
15 # interpreter executable name (csweb_nlm_static for nlm, csweb for standard
16 INTERPRETER=csweb.exe
17
18 # select server type. Standalone servers do not need a Web server to operate
19 # run on a separate port and that creates problems in cross-window scripting
20 # Mozilla-derived browsers
21 #SERVER=standalone
22 SERVER=fcgi
23
24 # the port to use if running in standalone server mode. Not used for fcgi ve
25 STANDALONEPORT=9999
26
27 # select storage system. Memory keeps all data in the memory of the server.
28 # the NCBI netcache system, and qman the NCBI queue manager system.
29 # For memory, configure the Web server not to run more than
30 # a single instance of the CGI, because only one instance has the state data
31 STORAGE=memcache
32 #STORAGE=memory
33 #STORAGE=netcache
34 #STORAGE=qman
35
36 # netcache/memcache service parameters
37 CACHEHOST=localhost
38 CACHEPORT=11211
39 #CACHEHOST=service1
40 #CACHEPORT=9001
41 #CACHESERVICE=NC_Test
42 #CACHESERVICE=NC_PCSketcher
43
44 # name of the host the application is running on. Full path name is required
45 # scripting.
46 HOST=xempc2.xemnet
47 #HOST=pubchemdev5.be-md.ncbi.nlm.nih.gov
```

It is recommended not to change the LOGDIR, DEBUG, INTERPRETER, STORAGE or SERVER variables except for advanced set-ups.

If this has all been configured correctly, store the edited Makefile in place and enter a shell command window. Make sure the current directory is the staging directory. There, type *make* and hit return once, thereby answering the question whether you want to regenerate the interface icons with a negative.

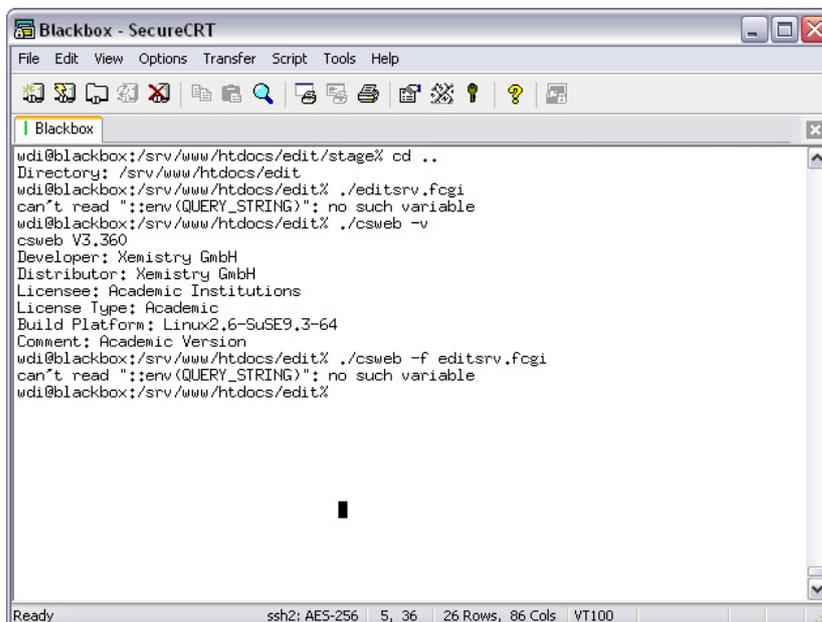
```
Blackbox - SecureCRT
File Edit View Options Transfer Script Tools Help
Blackbox
-e "s?@DEBUG?0?" \
-e "s?@PORT@?9999?" \
-e "s?@TRANSFER@?"smiles jme keys formula inchi"
? ? \
-e "s?@BGCOLOR@?#F0F0F0?" \
-e "s?@SELECTCOLOR@?#FFD700?" \
-e "s?@NEUTRALCOLOR@?#D8D8D8?" \
-e "s?@PATH@?/www/edit?" \
-e "s?@CGIPATH@?http://blackbox.xemnet/edit/edit"
srv.fcgi?" \
-e "s?@CACHEHOST@?localhost?" \
-e "s?@CACHEPORT@?11211?" \
-e "s?@CACHESERVICE@??" \
-e "s?@PGM@?csweb/;" \
fi; \
fi
rm -f /www/edit/csweb
if [ -x ./csweb ]; then \
  cp csweb /www/edit; \
else \
  cp `which csweb` /www/edit; \
fi
chmod +x /www/edit/csweb
chmod +x /www/edit/editorsrv
if [ "fcgi" = "fcgi" ]; then \
  mv /www/edit/editorsrv /www/edit/editorsrv.fcgi; \
  chmod +x /www/edit/editorsrv.fcgi; \
else \
  rm -f /www/edit/editorsrv.fcgi; \
fi
if [ "memcache" = "memcache" ]; then \
  rm -f /www/edit/memcached; \
  if [ -x ./memcached ]; then \
    cp memcached /www/edit; \
  else \
    cp `which memcached` /www/edit; \
  fi; \
  chmod +x /www/edit/memcached; \
fi
cp -r manual/HTML/Sketcher\ Manual/Output/* /www/edit/manual
mv /www/edit/manual/Frameset.html /www/edit/manual/index.html
rm -f /www/edit/Makefile
wdi@blackbox:/srv/www/htdocs/edit/stage% █
Ready ssh2: AES-256 43, 42 43 Rows, 86 Cols VT100
```

If all goes well, a few lines with lists of commands executed will scroll by, and the directory one level upwards – the base installation directory - is populated with files patched with the values of the Makefile variables.



The file explorer window above shows the contents of the Sketcher base installation directory after running the installation script.

Go one directory level upwards in the shell command window, and verify that the configured CGI script can be executed:



The error messages about the missing QUERY_STRING variable are what you should see at this time, since the software is run without CGI data. Any other message indicates a problem.

STEP 4: TESTING THE INSTALLATION

three *csweb* interpreters running in parallel. It does not matter which of the instances gets to service a specific client request, since the key in the request data will allow any of the instances to retrieve the state information from the store.

If your FCGI configuration has an automatic restart parameter which terminates the FCGI script interpreter and restarts it after a certain number of operations have been run on it, adjust the number to a rather high limit for this application, for example to 10000, or disable it completely. A low restart limit like 100 may be justified for standard CGI tasks executing for a number of seconds each, but in this context will lead to noticeable disruptions in the editing experience, since the sketcher generates a couple of events per second per user while the user is actively sketching. If an interpreter is forced to restart, no data loss occurs, but the start-up time can certainly be felt by the user.

If there are multiple sketcher hosts which are transparently accessed by clients in a rotating fashion, such as through a load balancer, the state caching mechanism must be centralized. The sketcher installation on all such hosts must point to the same cache host. If this is the case, host swapping will be handled gracefully and automatically. A sketcher which is set up with a localized *memcached* state store will attempt start the daemon on the local host, if it is not yet running, during the initialization of the interpreter. Obviously, this does not work for a remote host. In that case, the storage engine on that centralized host must be configured by means of *init* scripts, *inetd* configurations or other such mechanisms to either automatically start up and be permanently active, or be started on demand as soon as a request for the service arrives at the communication port. Multiple independent sketcher installations which do not swap hosts are free to use either a local store or a central set-up, or even multiple independent stores on a common host which are distinguished by a different communication port. Since state store keys are long and pseudo-random in character, there is no danger of colliding keys if multiple sketcher instances share a store.