

ECRITDHCP XPCOM Component

Overview

This is a XPCOM Component for use with Zap!, the Mozilla SIP Client based on XULRunner. The component is able to query DHCP options with the help of the operating system. The Windows version is developed with the help of Microsoft's DHCP functionality. The Linux version is still under development and for the moment depends on the output provided by the DHCP client software installed on the local machine.

Windows DLL

This component uses Windows functionality to collect DHCP options. In order to build successfully, Microsoft Visual C++ 2005 (Express) is needed with Microsoft Platform SDK. If you encounter problems with `atlthunk.lib` have a look at: http://developer.mozilla.org/en/docs/Mozilla_Build_FAQ Moreover, you need the following software from Mozilla: XULRunner SDK, Gecko SDK, Wintools and Moztools.

<http://releases.mozilla.org/pub/mozilla.org/xulrunner/releases/1.8.0.4/win32/en-US/xulrunner-1.8.0.4.en-US.win32.zip>

<http://releases.mozilla.org/pub/mozilla.org/xulrunner/releases/1.8.0.4/sdk/gecko-sdk-win32-msvc-1.8.0.4.zip>

<http://ftp.mozilla.org/pub/mozilla.org/mozilla/source/wintools.zip> (copy `glib-1.2.dll` and `libIDL-0.6.dll` from `wintools` to `gecko-sdk\bin`)

<http://ftp.mozilla.org/pub/mozilla.org/mozilla/libraries/win32/historic/vc8/vc8-moztools.zip>

For deployment, build a Release Version of the DLL files with Visual Studio (or modify the Manifest files accordingly). Otherwise the application cannot load the DLL files. Whenever you change the library files, force Zap to register the components again by deleting `compreg.dat` and `xpti.dat` from the `Mozilla\Zap` profiles folder in the User's Application Data folder. Sometimes it may also be necessary to increment the `BuildID` of the `application.ini` located in the `zap` application folder.

After compiling this project, copy the `ecritdhpc.dll` and the `ecritdhpc.xpt` files to the `zap` subfolder `xulrunner\components` manually.

If you encounter difficulties with `uuid.lib`, copy it to the Visual Studio project directory. `Uuid.lib` is located in the Microsoft Platform SDK folder.

Do not forget to alter the include path and library path settings for the project for your needs. The settings expect the project directory to be a subfolder of the XULrunner SDK (containing the Gecko SDK).

Linux Library

The Linux version of the ECRITDHCP contains a makefile. Apart from the C compiler, the Gecko SDK (version 1.7 works) is needed. Set the path to the Gecko SDK in the makefile (`GECKO_SDK_PATH`).

A Shared Library is created. Together with the `ecritdhpc.xpt` file it has to be copied as described in the Windows description. The Zap profile is located in `~/.mozilla/zap` under Linux.

The Linux version of the Library is still under development, since there is no standard way to request DHCP options with the help of the operating system. So, this library reads a file with the DHCP options created by the DHCP client `dhclient3`. Install `dhclient3` under your Linux

system and configure it the following way to get DHCP options 99 and 123 for location information.

```
/etc/dhcp3/dhclient.conf: add request for unknown-99 and unknown-123:
```

```
request subnet-mask, broadcast-address, time-offset, routers,  
       domain-name, domain-name-servers, host-name,  
       netbios-name-servers, netbios-scope, interface-mtu, nwip-suboptions, unknown-99,  
unknown-123;
```

```
/etc/dhcp3/dhclient-enter-hooks.d/debug: enable debug and write DHCP options to a file:
```

```
RUN="yes"
```

```
echo 99=$new_unknown_99 > /tmp/dhcp_loc.txt  
echo 123=$new_unknown_123 >> /tmp/dhcp_loc.txt
```

So, every time dhclient3 runs, the options are stored in /tmp/dhcp_loc.txt, where the XPCOM expects the file and reads the information and passes it on to Zap! Just call dhclient3 to refresh the DHCP options.

Interface

Currently, this ECRITDHCP XPCOM Component has the following interface:

```
#include "nsISupports.idl"
```

```
[scriptable, uuid(263ed1ba-5cc1-11db-9673-00e08161165f)]  
interface zapIdhcplib : nsISupports  
{  
    AString getDHCPoption(in long number);  
};
```

Javascript commands of Zap invoke the decode functionality of the library. As parameter for getDHCPoption, the number of the DHCP option is passed. As result, the value of the DHCP option is returned as HEX string.

The following code shows an example of how to retrieve DHCP options number 99 and 123 with the help of the XPCOM Component by Javascript commands in Zap:

```
const cid = "@enum.at/dhcp;1";  
var obj = Components.classes[cid].createInstance();  
obj = obj.QueryInterface(Components.interfaces.zapIdhcplib);  
dhcp_civic = obj.getDHCPoption(99);  
dhcp_geo = obj.getDHCPoption(123);
```

If this code results in an error message (“components.classes[cid] has no properties”), check the registration of the XPCOM component and force Zap to register again (as explained above).

More information on XPCOM is available at:

<http://www.mozilla.org/projects/xpcom/book/cxc/>
<http://starkravingfinkle.org/blog/2006/10/mozilla-platform-xpcom-in-c/>