webpsconsole

Web based PowerShell Console

The "Web based PowerShell Console" enables you to execute PowerShell Scripts and Cmdlets via a Web or Browser based PowerShell Console. Just install the tool on a mchine running IIS and use PowerShell via network!

See the corresponding article on my blog: http://ikarstein.wordpress.com/2011/09/02/webpsconsole-web-based-powershell-console-a-new-project-on-codeplex/

This project is a full featured Browser based PowerShell console that enables you to work on a server machine remotely.

It's like PowerShell remoting but it's not the same. While PowerShell remoting uses "WinRM" the tool that I've created uses a "normal" PowerShell host that will be executed in an IIS environment. (Please see my comment related to "PowerGUI Pro Mobile Shell" at the end of this article.)

That means: With my tool you will have a ASP.NET 4.0 based Web Application that can be accessed by a browser. The ASP.NET web app has a .ASPX site and some code behind. On the server in the ASP.NET context a PowerShell Host developed by me is running that accepts commands send by the clients Browser. If a command is send it will be executed by the PowerShell Host. All output is send back to the clients Browser.

It's on Codeplex: http://webpsconsole.codeplex.com with all source code! – It's ALPHA. I've done lots of testing but I'm a single developer with a single machine… I'd be happy to get your improvements or experiences!

Lets have a look at the app:

image

That is the "GUI". The black frame will contain the output of the server side PowerShell session.

Let's enter a command.

```
get-childitem c:\windows | select -first 5
```

image

After clicking "Send" the command is send to the server. The server executes the command and returns the PowerShell output.

Let's try this:

```
get-credential
```

image

You see the blue colored input box for the credential information input.

Here you see the implemented "session timeout":

image

After 15 minutes of inactivity you'll get a warning. After 20 minutes the session will be terminated. After termination you are not able to access the old session including session history!

Let's try this:

```
a$ = read-host
```

image

Here you get an input field for a single line of text. The output is stored in variable $a. Let's check the content of variable $a after sending the input:

image

Now let's test the functionality of completing missing cmdlet parameters:

```
get-content
```

image

image

Leave the last line empty and click "Send".

image

You get:

image

Now let's test the "choice" functionality:

image

After "Send" you get:

image

You see here the blue colored choice input field. If you choose "Halt Command" in this case the "Hello Ingo" command will not be executed:

image

If you select "Cancel" in such cases the complete PowerShell pipeline will be stopped! Not only the current command!

You can set colors a you need to:

image

Than "Send" and test it with:

image

You'll get:

image

Now let's have a look at the function "Download console as RTF". You'll find the link above the console frame.

image

Click "Open".

image

This is a Rich Text Format copy of the complete console output! – A "Clear-Host" will not clear this output! – This file can only be downloaded while the current session running. After the session ends you will not be able to access the information anymore!

Let's try "Show current buffer". This command is above the console frame too.

image

Here you get the "real" PowerShell buffer as HTML. This page can be save.

Let's try:

```
clear-host
```

image

Open "Show current buffer" again:

image

(It's empty now because clear-host did clear the PowerShell buffer!)

A word to the keyboard usability:

In the command input field you can use ESCAPE to clear the input field.

You can use CTRL+ENTER to send the command.

In other input fields (choice, credential, read-line,…) except "Read-Key" (see below) you can use ESCAPE to cancel the current operation and pipeline.

You can use there ENTER to send your entered data.

You can use TAB and SHIFT+TAB to navigate inside the input frame: input field, Send button and Cancel button.

The "Trace log" is an optional frame that maybe shows more information about your server connection and the "work behind". Just click the line and the frame will be shown:

image

("Send buffer as HTML" is caused by the "Show current buffer" function.)

You can close the PowerShell session by clicking the "x" in the dialog title (beside "v0.1.0.0"):

image

Now you can close the Browser or click "Ok" to start a new session. Or reload the Browser page to create a new session.

Setup

Now I want to tell you how to install the project.

First of all you need IIS. This can be installed on Windows 7 too.

You need a user account for executing PowerShell at server side. This account is used by every user of WebPSConsole: Each user of WebPSConsole will be impersonated at the server with the "execution account". **Here you should choose the account very carefully.**

1.

Create a directory on the server where you store the binaries. You create a folder "c:\inetput\webpsconsole".

image

2.

Copy the binaries there.

image

3.

Open IIS Manager.

Create a new Application Pool for the "execution account". This account need to execute ASP.NET **4.0** code.

image

Select "Classical Mode" for the application pool! – **Please check the settings.** In my case the settings were not used. After creating the app pool I had to edit it and set "Pipeline Mode" and "Framework Version" again!!!

4.

Create a new web application "WebPSConsole" using the previously created application pool.

Right click the "Sites" node in IIS Manager.

image

You get this dialog. Fill in your specific informations. Here is my sample. I'll use "ikWebPSConsole" as host header name in this setup demo because the host header "webpsconsole" already exists for development purpose.

image

Click the "Select" button beside "Application pool:

image

"OK".

My sample data:

image

(In order to get this working on my machine I have to edit c:\windows\system32\driver\etc\hosts and insert "ikWebPSConsole" as new local DNS entry.)

5.

Change the Authentication settings. Disable "Anonymous Access" and enable "Windows Authentication".

Select the web app in the treeview. On the right side double click "Authentication".

image

You get:

image

First double click "Anonymous Authentication" and deactivate it!

image

Now enable "Windows Authentication" the same way!

6.

You should restrict the usage of the web app by setting access restrictions.

image

There you can grant or deny access to the web app for specific users, e.g. administrators.

As an example: Here only "DOMAIN\Administrator" will have access to WebPSConsole.

image

You should enable SSL.

I've done this already in the setup example above while creating the web app.

You could use "SSL client certificates" to protect the web app.

6. Test the app.

Security

- I'ts as secure as you configure it!
- You can restrict the rights of the console on the server by selecting a carefully configured user account as application pool account. You don't have to configure this user to be "local admin" ;-)
- You can use SSL! (As shown above)
- You can use Client-SSL-Certificates!
- You can restrict client IP addresses!
- You can restrict access by secifying users (as shown above).
- BUT be sure you know what you do – as always ;-)

Limitations

- It's not as fast as a local PowerShell because of the client-server-interaction.
- Currently it runs only under the user context of the application pool. There should be "real" impersonation of the logged in user. But this did not work.
- It consumes memory on the server because of the data saved on the server in the session. Be sure to monitor the servers memory usage if you deploy it on live servers!
- There is not “Progress” support at the moment!

Ideas for future features:

- Command completion (like "Intellisence")
- Client side syntax check

- "Color code"

Please help me to impove the app! – Please post comment of your experiences!

Have fun!

For this project I've used:

- jQuery v1.6.2: http://www.jquery.com
- jQuery UI v1.8.15: http://www.jqueryui.com
- jQuery JSON v2.2: http://code.google.com/p/jquery-json/
- Json.NET 4.0 Release 2: http://json.codeplex.com/releases/view/64935

A word to PowerGUI Pro MobileShell

As I said I've developed this project in my last vacation. *After* finishing v0.1.0.0 I've seen **PowerGUI Pro MobileShell by Quest Software**
. It's based on the same idea as my project but is older and has more features I think. I only know a YouTube video of it because I do not know the commercial version of **PowerGUI**. – Beside this the free **PowerGUI** tool is my favorite PowerShell IDE. – For use on a live server you should think about using **PowerGUI Pro MobileShell**!