Simple Web Server, Using Python or PHP

If you need a quick way to test web pages and you don't want to go through the hassle (and overhead) of installing and configuring Apache, Python and PHP have web servers built right in!

Python 2

Change to the directory that contains your HTML files.

```
cd path/to/HTML/files
```

Start up the Python web server. This example will listen on port 8080, but you can use any port you want.

```
python -m SimpleHTTPServer 8080
```

Now, you access your files through http://localhost:8080.

You can also start an instance with CGI support. (The server will look for CGI scripts in path/to/HTML/files/cgi-bin by default)

```
python -m CGIHTTPServer 8080
```

I've only tested this with Python 2.x. My understanding is that SimpleHTTPServer and CGIHTTPServer are deprecated in Python 3 in favor of http.server.

Keep in mind that this only works for HTML/JavaScript files.

Python 3

Python 3 uses different syntax. After changing to the directory containing your HTML files, issue this command:

```
python3 -m http.server
```

If you need CGI support, use this:

```
python3 -m http.server --cgi
```

PHP

If you need support for server-side PHP programming, php.exe also has a web server built in. In the directory where your .php files reside, execute php.exe as follows:

```
php -S localhost:8080
```

Then, you can access your files through http://localhost:8080. Again, you can change the port to whatever you need.

If you need remote access to your files (from other machines on the local network), use this:

```
php -S 0.0.0.0:8080
```

Scripts

I put together a couple of scripts to simplify starting the server. (They both perform the same task. One is in bash script, the other is in Ruby. Use whichever you prefer.)

Bash version

localweb.sh

```
#!/usr/bin/env bash

statusmsg(){
    if [ $1 == "pysimple" ]; then
        echo "Simple Web Server, using Python"
    fi

    if [ $1 == "pycgi" ]; then
        echo "Simple Web Server w/ CGI Support, using Python"
    fi

    if [ $1 == "php" ]; then
        echo "Simple Web Server, using PHP"
    fi

    if [ $1 == "phprem" ]; then
        echo "Simple Web Server, using PHP, with remote access"
    fi

    echo "(Ctrl-C to exit the running server.)"
    echo ""
}
```

```
usage(){
    echo "USAGE:"
    echo ""
    echo "  $0 pysimple <port_number>   (starts a simple web server,
using Python)"
    echo "  or"
    echo "  $0 pycgi <port_number>      (starts a simple web server,
with CGI support, using Python)"
    echo "  or"
    echo "  $0 php <port_number>   (starts a simple web server, using
PHP)"
    echo "  or"
    echo "  $0 phprem <port_number>   (starts a simple web server,
using PHP, with remote access)"
    echo ""
    echo "e.g.: '$0 pycgi 81' starts a simple web server with CGI
support, using Python, listening on port 81."

    exit
}

if [ $# -ne 2 ]; then
    usage
fi

if [ $1 == "pysimple" ]; then
    statusmsg $1
    python -m SimpleHTTPServer $2
    exit
fi

if [ $1 == "pycgi" ]; then
    statusmsg $1
    python -m CGIHTTPServer $2
    exit
fi

if [ $1 == "php" ]; then
    statusmsg $1
    php -S localhost:$2
    exit
fi

if [ $1 == "phprem" ]; then
    statusmsg $1
    php -S 0.0.0.0:$2
    exit
fi
```

```
usage
```

Ruby version

localweb.rb

```
#!/usr/bin/env ruby

class WebMgr
    attr_accessor :server_type
    attr_accessor :server_port
    attr_accessor :usage_message

    def initialize
        @usage_message =
        "USAGE:\n" +
        "\tlocalweb pysimple <port_number>   (starts a simple web
server, using Python)\n" +
        "\tor\n" +
        "\tlocalweb pycgi <port_number>      (starts a simple web
server, with CGI support, using Python)\n" +
        "\tor\n" +
        "\tlocalweb php <port_number>        (starts a simple web
server, using PHP)\n" +
        "\tor\n" +
        "\tlocalweb phprem <port_number>     (starts a simple web
server, using PHP, with remote access)\n"

        if (ARGV[0] == nil)
            @server_type = ''
        else
            @server_type = ARGV[0]
        end

        if (ARGV[1] == nil)
            @server_port =  80
        else
            @server_port = ARGV[1]
        end
    end

    def StartServer
        case @server_type
            when 'pysimple'
                exec "python -m SimpleHTTPServer " + @server_port
            when 'pycgi'
                exec "python -m CGIHTTPServer " + @server_port
```

```
            when 'php'
                exec "php -S localhost:" + @server_port
            when 'phprem'
                exec "php -S 0.0.0.0:" + @server_port
            else
                puts @usage_message
        end
    end
end

### Main ###
begin
    myWebMgr = WebMgr.new

    if (ARGV[0] == nil)
        puts myWebMgr.usage_message
    else
        myWebMgr.StartServer
    end

rescue Exception => ex
    puts "An exception occurred: " + ex
end
```

Need More?

If you need more horsepower, i.e., an easy to configure LAMP dev server, I highly recommend XAMPP. This will give you a full LAMP stack that won't conflict with your existing setup.

ruby