Uživatel:ParcervenychBot/Zdrojový kód

crontab -e

```
# m h dom mon dow command
0 0 * * 1 python ~/Medicina/Wiki/bot.py 604800 #každé pondělí v 0:00, tj. spustí se každých 604800 vteřin
```

Soubor upload.php

```
<?php
if ($_POST['content'])
{
    $filename = "WikiskriptaCheck.txt";
    $file = fopen($filename, 'w') or die("Can't open file.");
    $text = $_POST['content'];
    $text = str_replace("\\'", "'", $text) //poměrně umělé řešení escape sekvence apostrofů
    fwrite($file, $text);
    fclose($file);
    echo "Upload success.";
}
else
    echo "Upload failure.";
    //die("Upload failure. POST methot error.");
?>
```

Soubor whitelist.php

```
<html>
    <head>
        <title>ParcervenychBot WhiteList</title>
        <meta http-equiv='Content-Type' content='text/html; charset=utf-8' />
    </head>
    <body>
    <?php
        if ($_POST['content'])
        {
            switch ($_POST['type'])
            {
                case "RedirectWhiteList":
                    $filename = "RedirectWhiteList.txt";
                    break;
                default:
                    $filename = "NPWhiteList.txt";
            }
            $file = fopen($filename, "a") or die("Can't open file: " . $filename);
            $link = $_POST['content'] . "\n";
            fwrite($file, $link);
            fclose($file);
            echo "Link added.";

        }
    if ($_POST['delete'])
    {
            if ($_POST['file'] == "RedirectWhiteList.txt")
                $filename = "RedirectWhiteList.txt";
            else
                $filename = "NPWhiteList.txt";

            $file = fopen($filename, "r");
            $newcontent = "";
            while( $entry = trim( fgets($file) ) )
            {
                if ($entry != $_POST['delete'])
            $newcontent .= $entry . "\n";
            }
            fclose($file);
            $file = fopen($filename, "w") or die("Can't open file: " . $filename);
            fwrite($file, $newcontent);
            fclose($file);
            echo "Link deleted.";
    }
    ?>
        <form action='<?php $_SERVER['php_self'] ?>' method='post'>
            <input type='text' name='content' /><br />
            <input type='radio' name='type' value='common' />NPWhiteList<br />
            <input type='radio' name='type' value='RedirectWhiteList' />RedirectWhiteList<br />
            <input type='submit' name='send' />
        </form>
        <hr />



        <h1>NPWhiteList.txt</h1>
        <ul>
    <?php

    $file = fopen("NPWhiteList.txt", "r");
    while( $entry = trim ( fgets($file) ) )
    {
        echo "\t\t\t<li>\n";
        echo "\t\t\t\t<form action='" . $_SERVER['php_self'] . "' method='post'>\n";
```

```
        echo "\t\t\t\t\t<input type='hidden' name='delete' value='$entry' />" . $entry . "\n";
        echo "\t\t\t\t\t<input type='hidden' name='file' value='NPWhiteList.txt' />\n";
        echo "\t\t\t\t\t<input type='submit' value='delete' />\n";
        echo "\t\t\t\t</form>\n";
        echo "</li>\n";
    }
    fclose($file)
    ?>
        </ul>
        <h1>RedirectWhiteList.txt</h1>
        <ul>
    <?php

    $file = fopen("RedirectWhiteList.txt", "r");
    while( $entry = trim( fgets($file) ) )
    {
        echo "\t\t\t<li>\n";
        echo "\t\t\t\t<form action='" . $_SERVER['php_self'] . "' method='post'>\n";
        echo "\t\t\t\t\t<input type='hidden' name='delete' value='$entry' />" . $entry . "\n";
        echo "\t\t\t\t\t<input type='hidden' name='file' value='RedirectWhiteList.txt' />\n";
        echo "\t\t\t\t\t<input type='submit' value='delete' />\n";
        echo "\t\t\t\t</form>\n";
        echo "</li>\n";
    }
    fclose($file)
    ?>
        </ul>

    </body>
</html>
```

bot.py

```
#coding=utf-8
import urllib, urllib2
import sys, time
import re

##############
#            #
#  nastaveni #
#            #
##############
user_name = ''
password = ''
#url = 'http://test-wiki.lf1.cuni.cz/api.php'
url = 'https://www.wikiskripta.eu/api.php'
urlWikiHome = 'https://www.wikiskripta.eu/index.php'
upload_server = 'http://myform.xf.cz/wiki/upload.php'
time_period = '' #pocet vterin do dalsiho spusteni bota, pokud je bot opakovane pousten zvenci, predava se tato informace pomoci sys.argv[1]
logfile_name = '' #název souboru, kam má bot ukládat svá hlášení, pokud prázdný, vypisuje do standardního výstupu
noportal_whitelist_link = 'http://myform.xf.cz/wiki/NPWhiteList.txt'




##############
#            #
#  funkce    #
#            #
##############
class LoginError(Exception):
    """Výjimka, jež bude vyvolána nepovedeným přihlášením k api serveru."""
    pass
class NoToken(Exception):
    """Výjimka, jež bude vyvolána, chybí-li token nutný k editaci."""
    pass
class EditError(Exception):
    """Výjimka, jež bude vyvolána nepovedenou editací."""
    pass




def get_the_POST_page(values, blank_values_list = [], apiserver = url):
    """Pošle požadavek html serveru 'apiserver' (implicitně wikiskripta) metodou POST s parametry 'values' typu dictionary a prečte si odpověd, kterou vrátí. Pokud je třeba metodou POST poslat proměnné, které nemají hodnoty, využije se 'blank_values_list' typu list."""
    data = urllib.urlencode(values)
    data2 = ""
    if blank_values_list:
        data2 = '&'.join(blank_values_list)
        data = data + '&' + data2
    request = urllib2.Request(apiserver, data)
    response = urllib2.urlopen(request)
    the_page = response.read()
    return the_page
def get_time(n):
    """Prevede cas (vteriny od zacatku epochy) do formatu hh:mm:ss yy/mm/dd. Pokud je argumentem n konstanta None, vraci tak aktualni cas."""
    now = time.localtime(n)
    actual_time = str(now[3]) + ':' + str(now[4]) + ':' + str(now[5]) + ' ' + str(now[0]) + '/' + str(now[1]) + '/' + str(now[2])
    return actual_time
def get_page_content(title):
    """Získá text wiki článku."""
    values = {"action": "query", "prop": "revisions", "rvprop": "content", "titles": title, "format": "txt"}
    content = get_the_POST_page(values)
    return content
def get_token(title = "Uživatel:ParcervenychBot"):
    """Získa edit token nutny k tomu, aby bylo mozne editovat clanky. Implicitne to ziskava pro stranku Uživatel:ParcervenychBot."""
    #values = {'action': 'query', 'prop': 'info|revisions', 'intoken': 'edit', 'titles': 'Speciální:Noportallink'}#neni mozne editovat specialni stranky
    values = {'action': 'query', 'prop': 'info|revisions', 'intoken': 'edit', 'titles': title}
```

```
    the_page = get_the_POST_page(values)
    edit_token = re.findall('(?<=edittoken=&quot;).*(?=&quot;)', the_page)
    logging("Edit token: " + edit_token[0])
    return edit_token[0]
def get_whitelist(listLink = noportal_whitelist_link):
    """Získá explicitně vytvořený seznam článků (txt soubor; názvy oddělené '\n') z adresy listLink."""
    request = urllib2.Request(listLink)
    response = urllib2.urlopen(request)
    wl_content = response.read()
    whitelist = wl_content.split('\n')
    return whitelist


def login(login_name = user_name, pswd = password):
    """Přihlásí uživatele login_name (implicitně user_name) k api serveru. Pokud není přihlášení úspěšné, vyvolá výjimku LoginError."""
    values = {'action': 'login', 'lgname': login_name, 'lgpassword': pswd}
    the_page = get_the_POST_page(values)
    if re.search('login result=&quot;Success&quot;', the_page):
        logging("Přihlášení proběhlo úspěšně.")
    else:
        logging("Nebylo možné se přihlásit. Zkuste změnit user_name nebo password.\n")
        raise LoginError, "Nebylo možné se přihlásit."
def logout():
    """Odhlasi uzivatele od api serveru."""
    values = {'action': 'logout'}
    get_the_POST_page(values)



def logging(string):
    """Pokud je nastavena hodnota log (implicitne logfile_name), uklada vystupy do souboru s timto jmenem, pokud ne, tiskne je do
standardniho vystupu."""
    global logfile
    global logfile_name
    if logfile_name:
        try:
            logfile.write(string + '\n')
        except NameError:
            logfile = open(logfile_name, "a")
            logfile.write(string + '\n')

    else:
        print string
def make_edit(new_content, summary = "aktualizace", title = "Uživatel:ParcervenychBot/Pískoviště"):
    """Na stranku title (implicitne Uživatel:ParcervenychBot) nahraje text new_content se shrnutim editace summary (implicitne
"aktualizace")."""
    global edit_token
    if not edit_token:
        raise NoToken, "Chybi edit token."
    logging("Nahrávání výsledku.")
    values = {'action': 'edit', 'title': title, 'text': new_content, 'summary' : summary, 'token': edit_token}
    the_page = get_the_POST_page(values)
    logging(the_page)
    if re.search('edit result=&quot;Success&quot;', the_page):
        logging("Nahravani vysledku probehlo uspesne.")
    else:
        logging("Nebylo mozne provest editaci:\n" + the_page)
        raise EditError, "Nebylo mozne provest editaci."
def make_output(noportaly, redirPortal, redJenOt, jenOtazky, pgs, podstranky, hesla, seznam_portalu, kategorie):
    """Vhodně zformátuje seznamy 'noportaly', 'pgs', 'podstranky' na vystup na wikiskripta. Ke každému článku vypíše jeho kategorie z
'kategorie' typu dictionary. Vrací řetězec ve wikikódu."""

    output = "{| class='infobox' style='border: 1px solid black; width: 450px; float: right'\n"
    output = output + "! style='background: lightsteelblue; text-align: center;' | Seznam všech portálů\n"
    for portal in seznam_portalu:
        output = output + "|-\n| [[" + portal + "]]\n"
    output = output + "|}\n"

    output = output + "== Stránky bez odkazu na porálu ==\n"
    output = output + "{| class='wikitable sortable'\n|-\n! Článek !! Kategorie\n"
    for title in noportaly:
        if title not in redirPortal and title not in redJenOt and title not in hesla and title not in pgs and title not in podstranky:
            if kategorie[title]:
                output = output + "|-\n| [[" + title + "]] || " + ", ".join(kategorie[title]) + "\n"
            else:
                output = output + "|-\n| [[" + title + "]] || &mdash;\n"
    output = output + "|}\n"

    output = output + "== Stránky s odkazem jen na portálu zkouškových otázek ==\n"
    output = output + "{| class='wikitable sortable'\n|-\n! Článek !! Kategorie\n"
    for title in jenOtazky:
        if title not in hesla and title not in pgs and title not in podstranky:
            if kategorie[title]:
                output = output + "|-\n| [[" + title + "]] || " + ", ".join(kategorie[title]) + "\n"
            else:
                output = output + "|-\n| [[" + title + "]] || &mdash;\n"
    output = output + "|}\n"

    output = output + "== Stránky postgraduálního studia bez odkazu na portálu ==\n"
    output = output + "{| class='wikitable sortable'\n|-\n! Článek !! Kategorie\n"
    for title in pgs:
        if kategorie[title]:
            output = output + "|-\n| [[" + title + "]] || " + ", ".join(kategorie[title]) + "\n"
        else:
            output = output + "|-\n| [[" + title + "]] || &mdash;\n"
    output = output + "|}\n"

    output = output + "== Podstránky bez odkazu na portálu ==\n"
    output = output + "{| class='wikitable sortable'\n|-\n! Článek !! Kategorie\n"
    for title in podstranky:
        if kategorie[title]:
            output = output + "|-\n| [[" + title + "]] || " + ", ".join(kategorie[title]) + "\n"
        else:
            output = output + "|-\n| [[" + title + "]] || &mdash;\n"
    output = output + "|}\n"

    output = output + "== Hesla bez odkazu na portálu ==\n"
    output = output + "{| class='wikitable sortable'\n|-\n! Článek !! Kategorie\n"
    for title in hesla:
```

```
        if kategorie[title]:
            output = output + "|-\n| [[" + title + "]] || " + ", ".join(kategorie[title]) + "\n"
        else:
            output = output + "|-\n| [[" + title + "]] || &mdash;\n"
    output = output + "|}\n\n"

    return output
def make_upload(text, url = upload_server):
    """K textu připojí vhodnou hlavičku a nahraje jej na server, na němž je komunikující php skript."""
    global time_period, time_per_ext
    logging("Probíhá upload.")

    header = '''<!-- KÓDOVÁNÍ UTF-8!!! -->
__NOTOC__
Tato stránka bude obdoba projektu [[w:cs:Wikipedie:WikiProjekt Check Wikipedia|WikiProjekt Check Wikipedia]]. Generuje ji pythonovský bot [[Uživatel:ParcervenychBot]], který vyhledává články, jež potřebují upravit (přidat odkaz na portálu, upravit html tagy, formátování nebo členění a podobně; zatím jen chybějící odkazy na portálu) a nahrává jejich seznam na http://myform.xf.cz/wiki/WikiskriptaCheck.txt
.\n\n'''
    posledni_aktualizace = get_time(None)
    header = header + "'"*3 + "Poslední aktualizace: " + posledni_aktualizace + "'"*3 + "\n\n"
    if time_period:
        aktualizace = time.time() + time_period
    if (time_per_ext) and (not time_period):
        aktualizace = time.time() + time_per_ext

    if time_period or time_per_ext:
        nasledujici_aktualizace = get_time(aktualizace)
        header = header + "'"*3 + "Následující aktualizace: " + nasledujici_aktualizace + "'"*3 + "\n\n"
        header = header + """Projektu můžete pomoci, pokud je současné datum pozdější než datum následující aktualizace. V tom případě
prosím:
*otevřete ve svém prohlížeči stránku http://myform.xf.cz/wiki/WikiskriptaCheck.txt
*nastavte kódování prohlížeče na &quot;Unicode (UTF-8)&quot; (Zobrazení &rarr; Kódování &rarr; Unicode (UTF-8) / View &rarr; Character
Encoding &rarr; Unicode (UTF-8) / či podobně)
*zkopírujte obsah stránky (např. Ctrl + A, Ctrl + C)
*editujte tuto stránku, nahraďte stávající text textem zkopírovaným, jako shrnutí editace napište &quot;aktualizace&quot; a uložte

Děkuji.

Případné přípomínky a náměty můžete psát do diskuze. Budu vděčný za zpětnou vazbu.

Jinak můžete pomoci Wikiskriptům tak, že článkům v následujících seznamech doplníte to, co postrádají (viz nadpis seznamu).\n\n"""
    else:
        header = header + "Můžete pomoci Wikiskriptům tak, že článkům v následujících seznamech doplníte to, co postrádají (viz nadpis
seznamu).\n\n"

    output = header + text
    values = {'content': output}
    the_page = get_the_POST_page(values, [], upload_server)
    if re.search('success', the_page):
        return "Upload úspěšný."
    else:
        return "Upload neuspesny."
    return output




def list_all_pages(namespace_number):
    """Vrací seznam všech článků wikiskript v zadaném jmenném prostoru. Pokud jmenný prostor není zadán, hledá v hlavním jmenném
prostoru."""

    if namespace_number:
        logging("Probíhá vyhledávání ve jmenném prostoru: " + str(namespace_number))
    else:
        logging("Probíhá vyhledávání v hlavním jmenném prostoru.")

    if namespace_number:
        values = {'action': 'query', 'list': 'allpages', 'apnamespace': namespace_number, 'aplimit': '500', 'apfilterredir':
'nonredirects'}
    else:
        values = {'action': 'query', 'list': 'allpages', 'aplimit': '500', 'apfilterredir': 'nonredirects'}
    the_page = get_the_POST_page(values)
    get_continues = re.compile('(?<=apfrom=&quot;).*(?=quot;)', re.UNICODE)
    continues = get_continues.findall(the_page)
    pages = the_page
    while continues:
        if namespace_number:
            values = {'action': 'query', 'list': 'allpages', 'apnamespace': namespace_number, 'aplimit': '500', 'apfrom': continues[0],
'apfilterredir': 'nonredirects'}
        else:
            values = {'action': 'query', 'list': 'allpages', 'aplimit': '500', 'apfrom': continues[0], 'apfilterredir': 'nonredirects'}
        the_page = get_the_POST_page(values)
        continues = get_continues.findall(the_page)
        pages = pages + "\n\n" + the_page
    get_titles = re.compile('(?<=title=&quot;).*(?=&quot;)', re.UNICODE)
    titles = get_titles.findall(pages)
    return titles
def list_noportaly(titles):
    """Vyhledá články ze seznamu titles, které nemají odkaz na portálu a články, které mají pouze odkaz v portálu zkouškových otázek.
Vrací dvojici (tuple) jejich seznamů. Funkce narocna na pocitaci cas api serveru."""
    global noportallinks_whitelist

    noportaly = []
    jenOtazky = []

    logging("Probíhá vyhledávání článků bez portálu.")
    get_portaly = re.compile('(?<=title=&quot;).*(?=&quot;)', re.UNICODE)
    get_otazky = re.compile('[Oo]tázky', re.UNICODE)
    for title in titles:
        values = {'action': 'query', 'list': 'backlinks', 'bltitle': title, 'blnamespace': '100', 'blfilterredir': 'all'}
        if title in noportallinks_whitelist: pass
        else:
            the_page = get_the_POST_page(values)
            portaly = get_portaly.findall(the_page)
            if portaly:
                otazky = True
                for portal in portaly:
                    if get_otazky.search(portal): pass
```

```
                    else: otazky = False
                if otazky: jenOtazky.append(title)
            else: noportaly.append(title)
    return (noportaly, jenOtazky)
def list_hesla(titles, kategorie):
    """Pomocí dictionary kategorie vyhledá články ze seznamu titles, které patří do kategorie hesla. Vrací jejich seznam."""
    hesla = []
    logging("Probíhá vyhledávání hesel.")
    for title in titles:
        try:
            if title in kategorie.keys():
                if "Heslo" in kategorie[title]:
                    hesla.append(title)
            else: logging("Stránka " + title + "nemá zjištěné kategorie.")
        except Exception as excp:
            print excp
            logging("Chyba na stránce " + title)
    return hesla
def list_PGS(titles):
    """Vyhledá články ze seznamu titles, které jsou články postgraduálního studia. Vrací jejich seznam."""
    pgs = []
    logging("Probíhá vyhledávání článků postgraduálního studia.")
    get_pgs = re.compile('/PGS', re.UNICODE)
    for title in titles:
        postgradual = get_pgs.search(title)
        if postgradual: pgs.append(title)
    return pgs
def list_podstranky(titles, pgs):
    """Vyhleda clanky ze seznamu titles, ktere jsou podstrankami, ale zaroven nejsou v seznamu pgs. Vraci jejich seznam."""
    podstranky = []
    logging("Probíhá vyhledávání podstránek.")
    get_podstranka = re.compile('/', re.UNICODE)
    for title in titles:
        if title not in pgs:
            podstranka = get_podstranka.search(title)
            if podstranka: podstranky.append(title)
    return podstranky
def list_pages_categories(titles):
    """Zjistí, do kterých kategorií jsou články zařazeny."""
    get_title = re.compile('(?<=title=&amp;quot;Kategorie:).*(?=&amp;quot;)', re.UNICODE)

    categories = {}
    logging("Probíhá vyhledávání kategorií.")
    for title in titles:
        values = {'action': 'query', 'prop': 'categories', 'titles': title}
        the_page = get_the_POST_page(values)
        categories_list = get_title.findall(the_page)
        categories[title] = categories_list
    return categories
def list_redirects(titles):
    """Vyhledá články, které sice nemají odkaz na portálu, ale ten mají stránky, které na ně přesměrovávají. Vrací jejich seznam. Kvůli
nesrovnalostem v API Wikiskript používá přímý dotaz na speciální stránku Speciální:Whatlinkshere"""
    global urlWikiHome
    redirects = []
    redJenOt = []
    logging("Probíhá vyhledávání přesměrujících článků.")
    get_redirects = re.compile('(?<=\>Portál:).*(?=\<)', re.UNICODE)
    get_otazky = re.compile('[Oo]tázky', re.UNICODE)
    for title in titles:
        request = urllib2.Request(urlWikiHome + '/Speciální:Whatlinkshere/' + title.replace(' ','_'))
        response = urllib2.urlopen(request)
        the_page = response.read()
        red = get_redirects.findall(the_page)
        jenOtazka = True
        if red:
            for each in red:
                if get_otazky.search(each): pass
                else: jenOtazka = False
            if jenOtazka: redJenOt.append(title)
            else: redirects.append(title)
    return (redirects, redJenOt)




#################
#               #
# telo programu #
#               #
#################

#inicializace
try:
    time_per_ext = int(sys.argv[1])
except IndexError:
    time_per_ext = ''

if logfile_name:
        logfile = open(logfile_name, "a")

noportallinks_whitelist = get_whitelist()

#vlastní cyklus
while True:
    start_time = time.time()
    now_time = get_time(None)
    logging("***" + now_time + "***")
    logging("Bot zacal plnit svuj ukol.")

    login()


    all_pages = list_all_pages(None)
    all_portaly = list_all_pages(100)
    (noportaly,jenOtazky) = list_noportaly(all_pages)
    (redirPortal, redJenOt) = list_redirects(noportaly)
    jenOtazky = jenOtazky + redJenOt
    pages_categories_map = list_pages_categories(noportaly + jenOtazky)
```

```
    hesla = list_hesla(noportaly + jenOtazky, pages_categories_map)
    pgs = list_PGS(noportaly + jenOtazky)
    podstranky = list_podstranky(noportaly + jenOtazky, pgs)

    edit_token = get_token()
    output = make_output(noportaly, redirPortal, redJenOt, jenOtazky, pgs, podstranky, hesla, all_portaly, pages_categories_map)

    #make_edit(output)
    upload_status = make_upload(output)
    logging(upload_status)

    logout()

    end_time = time.time()
    last = int(end_time - start_time)
    logging("Bot provadel praci po dobu " + str(last) + " vterin (" + str(last/60) + " minut).")
    if time_period == 0 or time_period == '':
        break
    new_start = time_period - last
    logging("Mel by zacit znovu za " + str(new_start) + " vterin. Muzete jeho proces ukoncit stisknutim klaves Ctrl+C.")
    time.sleep(new_start)


try:
    logfile.close()
except NameError:
    pass
```