

# Списание .KOM

WebDev Bulgaria Magazine

издание за територията на България

брой 3 /2005

списание за интернет технологии списание за интернет технологии списание за интернет технологии списание за интернет

Python

Брой 3/2005



Съдържание

```
class CountryByIP(IPRangeDB):
    sources = {
        'arin' : ('ftp://ftp.arin.net/pub/stats/arin/', 'arin.%Y%m%d'),
        'ripencc': ('ftp://ftp.ripe.net/ripe/stats/', 'ripencc.%Y%m%d'),
        'apnic' : ('ftp://ftp.apnic.net/pub/stats/apnic/', 'apnic-%Y-%m-%d')
    }
```

- PHP: Класове  
- PERL: E-mail



- Python: IP  
- XML: Mozilla

"Никога не е късно да се започне изучаването на най-прекрасното нещо в програмирането  
...XML"

списание .KOM

powered by OIC Freedom

списание за интернет технологии

списание за интернет технологии

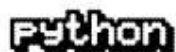
<http://www.spisanie.com> bogomil@spisanie.com

powered by OIC Freedom

## Съдържание



- 4. Работа с класове: 'Кой ме вика'
- 5. PHP: Извикване на обратния път



- 8. Python: Определяне на държавата по IP адреса



- 11. Изпращане на поща с Perl



- 15. Писане на плъгини за Мозила (2)

Този брой **компилираха** за вас:

**Богомил Шопов**

Издател – OIC  
[bogomil@spisanie.com](mailto:bogomil@spisanie.com)

**Христина Иванова**

Мениджър контакти – OIC  
[hristina@spisanie.com](mailto:hristina@spisanie.com)

Съдържание и дизайн

**OIC**

[office@openideascompany.com](mailto:office@openideascompany.com)  
<http://openideascompany.com>

**Лого** на списанието



Издател:



Вземете и брой 1 на OpenSourceMag



брой 1/2005

<http://spisanie.cc>

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)

powered by **OIC**  
**Freedom**

## Top5

Класацията ТОП 5 е на редакторите на "Списание точка ком" и отчита тяхното виждане за развитието на интернет технологиите у нас. Могат да бъдат предлагани всякакви сайтове, продукти, услуги свързани с интернет развитието, технологии, култура и т.н.

### В края на годината, от всички победители, ще се избере идеалната петица

Ето как изглежда сегашната класация:



- 1) **[.PHP-BG](http://php-bg.org)** - сайт за PHP, дискусии и много други дейности, свързани с този популярен уеб език. Сайтът вече е с нова версия и предлага по-добра информация.  
<http://php-bg.org>



- 2) **[.PHPGuru](http://phpguru.digitalniagolov.com/start/index.php)** - Още един сайт за PHP, който за жалост не се поддържа често, но има много добра основа за развитие в добра посока.  
<http://phpguru.digitalniagolov.com/start/index.php>

- 3) **[.Firebird](http://www.fbtalk.net/viewforum.php?id=28)** - Форум за тази много добра база данни на български.  
<http://www.fbtalk.net/viewforum.php?id=28>



- 4) **[.Clever Developers](http://cleverdevelopers.blogspot.com/)** - полезен блог посветен на Firebird и на интеграцията му в различни приложения.  
<http://cleverdevelopers.blogspot.com/>

- 5) не се присъжда



## Работа с класове: 'Кой ме вика'

Как да разберем от кой клас идва заявката към друг клас на PHP5? Това обикновено не е много трудно, но малко хора го знаят.

Нека да имаме следната схема:

- Имаме клас **DBQuery**, който има метод *execute*
- *Имаме клас **TestDB**, който използва метода на горния клас.*

### PHP

```
class TestDB
{
function sqlquery()
{
    DBQuery::execute();
}
}

class DBQuery
{

function execute()
{

echo "mene me tyrsi: ".get_class($this);

}

}

$a= new TestDB;
$a->sqlquery();
```

Какво, обаче може да се случи тук? Нека да разгледаме следния пример:

## PHP

```
class TestDB
{
    function sqlquery()
    {
        DBQuery::execute();
    }
}

class DBQuery
{
    function execute()
    {
        echo "mene me tyrsi: ".get_class($this);
    }
}

TestDB::sqlquery();
```

Можете да видите каква е точно разликата между двата кода, които не се различават много по логиката.



## PHP: Извикване на обратния път

Понякога се налага да имаме достъп до информацията, къде точно е минало управлението на един скрипт. Ако имаме 10 000 класове в добре да знаем как точно и през кои от тях е станало изпълнението на логиката.

Това е необходимо, ако искаме да си направим debug на кода, както и за много други дейности, като статистика, обработка на грешките или просто поради любопитство.

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)

PHP предлага една такава функция `debug_backtrace()`; ,която можем да използваме.

Докато подготвяхме тези материали, попаднах на една много интересна функция, построена на основата на горната.

Нека да я видим:

## PHP

```
function backtrace()
{
    $output = "<div style='text-align: left; font-family: monospace;'>\n";
    $output .= "<b>Backtrace:</b><br />\n";
    $backtrace = debug_backtrace();

    foreach ($backtrace as $bt) {
        $args = "";
        foreach ($bt['args'] as $a) {
            if (!empty($args)) {
                $args .= ', ';
            }
            switch (gettype($a)) {
                case 'integer':
                case 'double':
                    $args .= $a;
                    break;
                case 'string':
                    $a = htmlspecialchars(substr($a, 0, 64)).((strlen($a) > 64) ? '...' : '');
                    $args .= "\"$a\"";
                    break;
                case 'array':
                    $args .= 'Array(' . count($a) . ')';
                    break;
                case 'object':
                    $args .= 'Object(' . get_class($a) . ')';
                    break;
                case 'resource':
                    $args .= 'Resource(' . strstr($a, '#') . ')';
                    break;
                case 'boolean':
                    $args .= $a ? 'True' : 'False';
                    break;
                case 'NULL':
                    $args .= 'Null';
                    break;
                default:
                    $args .= 'Unknown';
            }
        }
        $output .= "<br />\n";
        $output .= "<b>file:</b> {$bt['line']} - {$bt['file']}<br />\n";
        $output .= "<b>call:</b> {$bt['class']}{$bt['type']}{$bt['function']}($args)<br />\n";
    }
    $output .= "</div>\n";
    return $output;
}
```

Тъй като тази функция, сама по себе си, едва ли ви говори нещо сама за себе си, нека да я видим в действие.

1) Направете един файл, наречен **trace.php** и запишете тази функция вътре.

2) Отворете написания в предната статия файл (този с класовете) и го сложете в метода *execute()* на DBQuery, така, че да се получи следното:

## PHP

```
...  
class DBQuery  
{  
  
    function execute()  
    {  
        echo backtrace();  
        echo "mene me tyrsi: ".get_class($this);  
    }  
  
}  
....
```

3) Изпълнете файла. Резултатът би трябвало да е следния:

## Резултат

```
Backtrace:  
file: 22 - c:\Apache\docstestclass.php  
call: backtrace()  
file: 10 - c:\Apache\docstestclass.php  
call: DBQuery::execute()  
file: 29 - c:\Apache\docstestclass.php  
call: TestDB::sqlquery()  
mene me tyrsi:
```

или

## Резултат

```
Backtrace:  
file: 22 - c:\Apache\docstestclass.php  
call: backtrace()  
file: 10 - c:\Apache\docstestclass.php  
call: TestDB->execute()  
file: 30 - c:\Apache\docstestclass.php  
call: TestDB->sqlquery()  
mene me tyrsi: TestDB
```



## Python: Определяне на държавата по IP адреса

Данните за регистрация на IP адресите се съхраняват в бази данни whois. За да се предостави възможност за анализиране на трафика, **RIPE NCC**, **ARIN** и **APNIC** един път месечно правят извлечение на своите бази данни. Именно тези данни ще използваме за да съставим наша локална такава.

За начало, обаче, трябва да организираме всичките тези диапазони от IP адреси за да имаме много бърз достъп до тях. Нека да използваме Btree на базата BerkleyDB, достъпа до които ще се осигури от функцията:

### Python

**btopen()** от стандартния модул bsddb.

### Python

```
from bsddb import btopen
from socket import inet_aton, inet_ntoa

class IPRangeDB:

    def __init__(self, filename, mode='r'):
        self.__db = btopen(filename, mode)

    def close(self):
        self.__db.close()

    def _locate(self, ip):
        db = self.__db
        try:
            first, record = db.set_location(ip)
        except KeyError:
            try:
                first, record = db.last()
            except KeyError:
                raise KeyError(inet_ntoa(ip))
        else:
            if first != ip:
                first, record = db.previous()
        assert first <= ip
        return first, record

    def __getitem__(self, ip_str):
        ip = inet_aton(ip_str)
        first, record = self._locate(ip)
        last = record[:4]
```



```
assert last>=first
if ip<=last:
    return self.unpack(record[4:])
else:
    raise KeyError(ip_str)

def add(self, first_str, last_str, info):
    first = inet_aton(first_str)
    last = inet_aton(last_str)
    try:
        db_first, record = self._locate(last)
    except KeyError:
        pass
    else:
        db_last = record[:4]
        if first<=db_last:
            raise ValueError(
                'Range %s-%s intersects ' % (first_str, last_str) +
                'with existing entry %s-%s' %
                (inet_ntoa(db_first), inet_ntoa(db_last)))
        self.__db[first] = last+self.pack(info)

def pack(self, info):
    return info

def unpack(self, info):
    return info
```

Методът `_locate()` търси запис с максимално близък или съвпадащ IP адрес, предаван й като параметър. Методът `__getitem__()` позволява да се използва класът `IPRangeDB` аналогично

### Python

```
>>> db = IPRangeDB('test.db', 'c')
>>> db.add('10.0.0.0', '10.255.255.255', 'Наша локална сеть')
>>> print db['10.1.2.3']
Наша локална сеть
>>> print db['123.45.67.89']
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
  File "ip2cc.py", line 38, in __getitem__
    raise KeyError(ip_str)
KeyError: 123.45.67.89
```

Остана само да намерим начин за напълване на базата данни:

### Python

```
from urllib import urlopen
from xreadlines import xreadlines
```

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)

```
from time import strptime
import struct

class CountryByIP(IPRangeDB):

    sources = {
        'arin' : ('ftp://ftp.arin.net/pub/stats/arin/', 'arin.%Y%m%d'),
        'ripenc' : ('ftp://ftp.ripe.net/ripe/stats/', 'ripenc.%Y%m%d'),
        'apnic' : ('ftp://ftp.apnic.net/pub/stats/apnic/', 'apnic-%Y-%m-%d')
    }

    def fetch(self):
        for name in self.sources:
            fp = self.__openRecent(name)

            for line in xreadlines(fp):
                parts = line.strip().split('|')
                if len(parts)==7 and parts[2]=='ipv4' and \
                    parts[6] in ('allocated', 'assigned') and \
                    name==parts[0]:
                    first = parts[3]
                    first_int = struct.unpack('!i', inet_aton(first))[0]
                    last_int = first_int+int(parts[4])-1
                    last = inet_ntoa(struct.pack('!i', last_int))
                    try:
                        self.add(first, last, parts[1].upper())
                    except ValueError:
                        pass

    def __openRecent(self, name):
        uri, format = self.sources[name]
        files = []
        for line in xreadlines(urlopen(uri)):
            file = line.split()[-1]
            try:
                dt = strptime(file, format)
            except ValueError:
                pass
            else:
                files.append((dt, file))
        files.sort()
        return urlopen(uri+files[-1][1])
```

Методът `__openRecent` намира последните санни и връща файлов обект. Методът `fetch` анализира данните, избира необходимите такива и ги въвежда в базата. Използването на модула `xreadlines` позволява да се анализират данните по ред на постъпване.

## Python

```
>>> db = CountryByIP('test.db', 'n')
>>> db.fetch()
```

```
и използватъ  
>>> from socket import gethostbyname  
>>> db[gethostbyname('python.org')]  
'NL'
```



## Изпращане на поща с Perl

Много често при програмиране на какъвто и да е език, стои въпроса за изпращане на електронна поща. Това, разбира се е така и при Perl. Целта на този материал е да разгледаме няколко начина как може да стане това

Повечето от вас сигурно предпочитат този начин:

### Perl

```
open(MAIL,"| /usr/sbin/sendmail test@test.com");  
print MAIL "From: from\@test.com\n";  
print MAIL "Subject: А сега да видим\n\n";  
print MAIL "Тествам си само \n";  
close(MAIL);
```

Този метод, обаче не винаги върши работа във всички ситуации. Нека да хвърлим едно око на готовите модули за Perl

## Модул - MIME::Lite

Този модул позволява лесно и бързо да създаваме и изпращаме e-mail във формат в какъвто искаме. След като си го свалим от [cpan.org](http://cpan.org), инсталацията му е много лесна

```
perl Makefile.PL  
make test  
make install
```

Когато модула е инсталиран, можем да го изтестваме за да видим как работи. За начало нека да изпратим писмо в plain текст:

## Perl

```
#!/usr/bin/perl

use MIME::Lite;

my $msg = MIME::Lite->new( From    => 'foo@test.com',
                           To      => 'bar@test.com',
                           Subject => 'Пак тестваме сета',
                           Type    => 'text/plain; charset=windows-1251',
                           Data    => 'това е пратено с новия модул' );

$msg->send();
```

Можем да видим от горния код, че това става много много лесно. Нека сега да направим и по-сложен пример, като изпратим писмо с прикачен файл.

## Perl

```
#!/usr/bin/perl

use MIME::Lite;

my $msg = MIME::Lite->new( From    => 'foo@test.com',
                           To      => 'bar@test.com',
                           Subject => 'Писмо с прикачен файл',
                           Type    => 'multipart/mixed' );

$msg->attach( Type => 'text/plain; charset=windows-1251',
             Data => "Ето писмото с картинката." );

$msg->attach( Type    => 'image/gif',
             Path     => 'mylogo.gif',
             Filename => 'logo.gif',
             Disposition => 'attachment' );

$msg->send();
```

Както виждате модулет позволява да се създават писма с с отделни части:

**Определяме параметрите на самото писмо**  
**Добавяме частите му, в случая първо текста**

## Perl

```
$msg->attach( Type => 'text/plain; charset=windows-1251',  
  Data => "Ето писмото с картинката." );
```

А след това и прикачения файл.

## Perl

```
$msg->attach( Type      => 'image/gif',  
  Path        => 'mylogo.gif',  
  Filename    => 'logo.gif',  
  Disposition => 'attachment' );
```

Хайде сега и да изпратим още по сложно писмо. Това не значи, че кода ще е много по сложен от първите два. Нека сега да изпратим html файл с графика, прикрепен zip файл, но и да има версия за клиенти, които не могат да четат html формат.

## Perl

```
#!/usr/bin/perl  
  
use MIME::Lite;  
  
my $msg = MIME::Lite->new( From   => 'foo@test.com',  
  To      => 'bar@test.com',  
  Subject => 'Сложното писмо',  
  Type    => 'multipart/mixed' );  
  
$msg->attach( Type => 'text/html; charset=windows-1251',  
  Data => "<body>\n<h1>Ето какво ти пращам</h1>\n<br><br>\nИма и картинка:<br><img src=\"cid:logo.gif\">\n</body>\n" );  
  
$msg->attach( Type => 'text/plain; charset=windows-1251',  
  Data => "Това го четете ако няма HTML.\n" );  
  
$msg->attach( Type      => 'image/gif',  
  Path        => 'mylogo.gif',  
  Filename    => 'logo.gif',  
  Id          => 'logo.gif',  
  Disposition => 'attachment' );  
  
$msg->attach( Type      => 'application/x-zip-compressed',  
  Path        => 'price.zip',  
  Filename    => 'price.zip',  
  Disposition => 'attachment' );  
  
$msg->send();
```

С помощта на този модул можем лесно и бързо да изпратим e-mail в какъвто формат искаме.

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)

Може би сте забелязвали, че понякога пристигат писма, на които не можете да прочетете темата, а самото писмо е четимо. За да се промени това, можем да ползваме следния модул MIME::Base64.

```
...  
$subj = MIME::Base64::encode("Тема писъма","");  
$subj = "=?windows-1251?B?".$subj."?=";  
...
```

За да изпратим писмо с MIME::Lite и MIME::Base64 , то би трябвало да изглежда така:

### Perl

```
#!/usr/bin/perl  
  
use MIME::Lite;  
use MIME::Base64;  
  
my $subj = MIME::Base64::encode("А сега да видим","");  
$subj = "=?windows-1251?B?".$subj."?=";  
  
my $msg = MIME::Lite->new( From => 'foo@test.com',  
                           To   => 'bar@test.com',  
                           Subject => $subj,  
                           Type   => 'multipart/mixed' );  
  
$msg->attach( Type => 'text/html; charset=windows-1251',  
             Data => "<body>\n<h1>Тук ти пиша</h1>\n<br><br>\n...със картинка:<br><img src=\"cid:logo.gif\">\n</body>\n" );  
  
$msg->attach( Type => 'text/plain; charset=windows-1251',  
             Data => "Ако нямаш HTML поддръжка.\n" );  
  
$msg->attach( Type   => 'image/gif',  
             Path    => 'mylogo.gif',  
             Filename => 'logo.gif',  
             Id       => 'logo.gif',  
             Disposition => 'attachment' );  
  
$msg->attach( Type   => 'application/x-zip-compressed',  
             Path    => 'price.zip',  
             Filename => 'price.zip',  
             Disposition => 'attachment' );  
  
$msg->send();
```



## XML: Писане на плъгин посредством XML

Тъй като в един от предните броеве, бяхме обещали да ви покажем как се правят плъгини за мозила, ето , че и това време дойде. За да може да работи каквото и да е било на много места, то трябва да е направено въз основата на някакъв стандарт. Плъгините на мозила са базирани на XML стандарт Sherlock разработен от Apple.

Ето как трябва да бъде 'построен ' един такъв плъгин за да може да работи

### 1) Коментари и описание

```
# Spisanie
# Mozilla plugin for spisanie.com (bulgarian language)
# by Bogomil Shopov
#
# Created: February 24, 2005
# Latest update: February 24, 2005
#
# Status: Working
```

### 2) Основна част с инструкции, където се описва името , къде се извършва търсенето и как да се покажат резултатите, ако има успех.

```
<search
version="7.1"
name="име на плъгина"
description="описание"
action="къде се изпълнява търсенето"
searchForm="къде се намира формата за търсене"
method="метод">

<input name="sourceid" value="Mozilla-search">
<input name="keywords" user="">

<interpret
resultListStart="<!-- маркер за начало на списък -->"
resultListEnd="<!-- маркер за край на списък -->"
resultItemStart="<!-- маркер за начало на съдържание -->"
resultItemEnd="<!-- маркер за край на съдържание -->"

</search>
```

### 3) Инструкции за браузера

```
<browser
```

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)

```
update="URL за опресняване"  
updateIcon="икона на пългина"  
updateCheckDays="период">
```

Сега нека да напишем един истински пългин за търсене за да можем да проверим знанията си в реална обстановка;)

```
<search  
version="7.1"  
name="Linux-bg.org"  
description="Linux for Bulgarians"  
action="http://linux-bg.org/cgi-bin/y/index.pl"  
searchForm="http://linux-bg.org/cgi-bin/y/index.pl"  
queryEncoding="windows-1251"  
queryCharset="windows-1251"  
method="GET">  
  
<input name="act" value="big_search">  
<input name="page" value="search">  
<input name="keywords" user="">  
  
<browser  
update="http://...."  
updateIcon="linuxbg.gif"  
updateCheckDays="12">
```

Ако искате да намерите много пългини или да предложите свой за слагане към сайта, можете а отидете тук:

<http://mycroft.mozdev.org/>

Там са сложени вече и около 4 български пългини, освен това, бета версиите на много такива има и тук: <http://spisanie.com/addPlugins.html>

## Google + ID

Нека да погледнем и как става интеграцията с Google и клиентско ID. Тоест, ако искате да имате търсачка в Google само за сайта ви, това може да стане така.

## XML

```
# BgLog  
# Mozilla plugin for BgLog.net /Google (bulgarian language)  
#  
#  
# Created: May 28, 2005  
# Latest update: May 28, 2005  
# Licence: GPL v2  
# Status: Working  
  
<search  
name="BGLog"  
description="BgLog - bulgarian blogger community"
```

<http://www.spisanie.com> [bogomil@spisanie.com](mailto:bogomil@spisanie.com)



```
method="GET"
action="http://www.google.com/custom"
queryEncoding="utf-8"
queryCharset="utf-8">

<input name="q" user>
<input name="sourceid" value="mozilla-search">
<input name="domains" value="www.bglog.net">
<input name="sitesearch" value="www.bglog.net">
<input name="client" value="pub-5777067443749375">
<input name="forid" value="1">
<input name="ie" value="UTF-8">
<input name="oe" value="UTF-8">
<input name="cof"
value="GALT:#008000;GL:1;DIV:#336699;VLC:663399;AH:center;BGC:FFFFFF;LBGC:33669
9;ALC:0000FF;LC:0000FF;T:000000;GFNT:0000FF;GIMP:0000FF;FORID:1;">
<input name="hl" value="bg">

<interpret
  browserResultType="result"
  charset = "UTF-8"
  resultListStart="<!--a-->"
  resultListEnd="<!--z-->"
  resultItemStart="<!--m-->"
  resultItemEnd="<!--n-->"
>
</search>
<browser
update="http://spisanie.com/bglog.src"
updateIcon="http://spisanie.com/bglog.png"
updateCheckDays="15">
```

## Site:

Още по интересно е ако искате да търсите директно в Google, използвайки **Site: условието**

```
# Mozilla plugin for Maguma GmbH
#
# SearchSite: Maguma (www.maguma.com)
#
#
# Created: June 20, 2005
# Latest update: June 20, 2005
# Licence: GPL v2
# Status: Working

<search
version="7.1"
name="Maguma"
description="The best PHP IDE"
action="http://www.google.com/search"
searchForm="http://www.google.com/search"
queryEncoding="utf-8"
queryCharset="utf-8"
```

<http://www.spisanie.com> bogomil@spisanie.com

```
method="GET">

<input name="source" value="mozilla-search">

<input name="q" value="site:maguma.com">
<input name="q" user="">
<inputnext name="start" factor="10">
<inputprev name="start" factor="10">
<input name="ie" value="utf-8">
<input name="oe" value="utf-8">

<interpret
  browserResultType="result"
  charset = "UTF-8"
  resultListStart="<!--a-->"
  resultListEnd="<!--z-->"
  resultItemStart="<!--m-->"
  resultItemEnd="<!--n-->"
>
</search>

# This is a test only, please do not send it to mozilla yet
<browser
update="http://mycroft.mozdev.org/plugins/open-culture.src"
updateIcon="http://mycroft.mozdev.org/plugins/open-culture.jpg"
updateCheckDays="3">
```

Това е наистина интересно решение, понеже го няма документирано в насоките на Мозила за правенето на плъгини:

## XML

```
<input name="q" value="site:името на сайта">
<input name="q" user="">
```

## Полезни линкове

<http://spisanie.com>

- Новини, форуми и изненади. Тук ще намерите информация за всички броеве на списанието и много документи.

<http://spisanie.cc>

- Списание за отворен код, философия и култура. Можете да го свалите безплатно още сега.

<http://openideascompany.com>

- Сайт на компанията за отворени идеи. Информация за продукти и услуги, както и за свободни работни места.

<http://www.spisanie.com> bogomil@spisanie.com

powered  
by **Freedom**