

## eZ Components

PHP Norge Medlemsmøte - Oslo, Norway

Derick Rethans - dr@ez.no

<http://files.derickrethans.nl/ezc-php-norge.pdf>

# Development Goals

- Move to PHP 5.1.1
- Thoroughly plan the product to get a top notch API for the library
- Thoroughly tested. All code should be covered by unit tests prior to implementation (PHPUnit)
- Make sure we keep the product open enough for future development
- Do proper documentation during the development

- Provide a solid platform for PHP application development
- Don't force a structure: no "framework"
- Clean and simple API
- Excellent documentation
- Keep backward compatibility for longer periods of time
- Stable and few regressions
- Clean IP, Open Source friendly

## Documentation:

- Makes a library usable
- API documentation
- Tutorials
- Example Applications
- <http://ez.no/doc/components/overview>

## PHP Documentor:

- Made for PHP
- In use by PEAR
- Supported by PHP IDEs

# License and Contributing

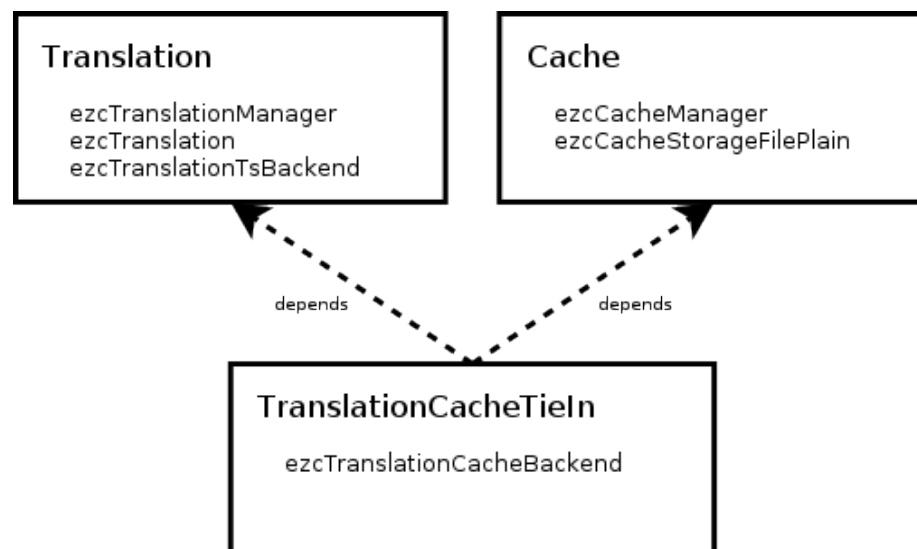
## Clean IP, Open Source friendly

- New BSD license: Open Source, very permissive, compatible with GPL
- CLA: To accept contributions to the components we need a signed "Contributor Licensing Agreement"

# Dependencies

- The less the better
- Only if really necessary
- Dependency-only packages

Tie-Ins:



- Pre-fixed: for namespacing
- Readable: for sanity
- Slightly Mangled: for clarity

ezcMail vs. Mail

ezcTestSuite vs. PHPUnit2\_Framework\_TestSuite

ezcMailSmtptTransport vs. ezcMailSMTPTransport

```
<?php
require 'ezc/Base/base.php';
function __autoload( $className )
{
    ezcBase::autoload( $className );
}

$a = new ezcMailSmtptTransport( 'localhost' );
var_dump( $a );
?>
```

Using the classname's part as path elements makes ugly paths:

```
ezcMailParserSet      => Mail/parser/set.php,  
ezcMailSmtptTransport => Mail/smtpt/transport.php,  
ezcMailTransportOptions => Mail/transport/options.php,  
ezcMailAddress        => Mail/mail/address.php,
```

More logical names (mail\_autoload.php):

```
ezcMailParserSet      => Mail/parser/interfaces/parser_set.php,  
ezcMailSmtptTransport => Mail/transports/smtpt/transport_smtpt.php,  
ezcMailTransportOptions => Mail/options/transport_options.php,  
ezcMailAddress        => Mail/structs/mail_address.php,
```

Some problems:

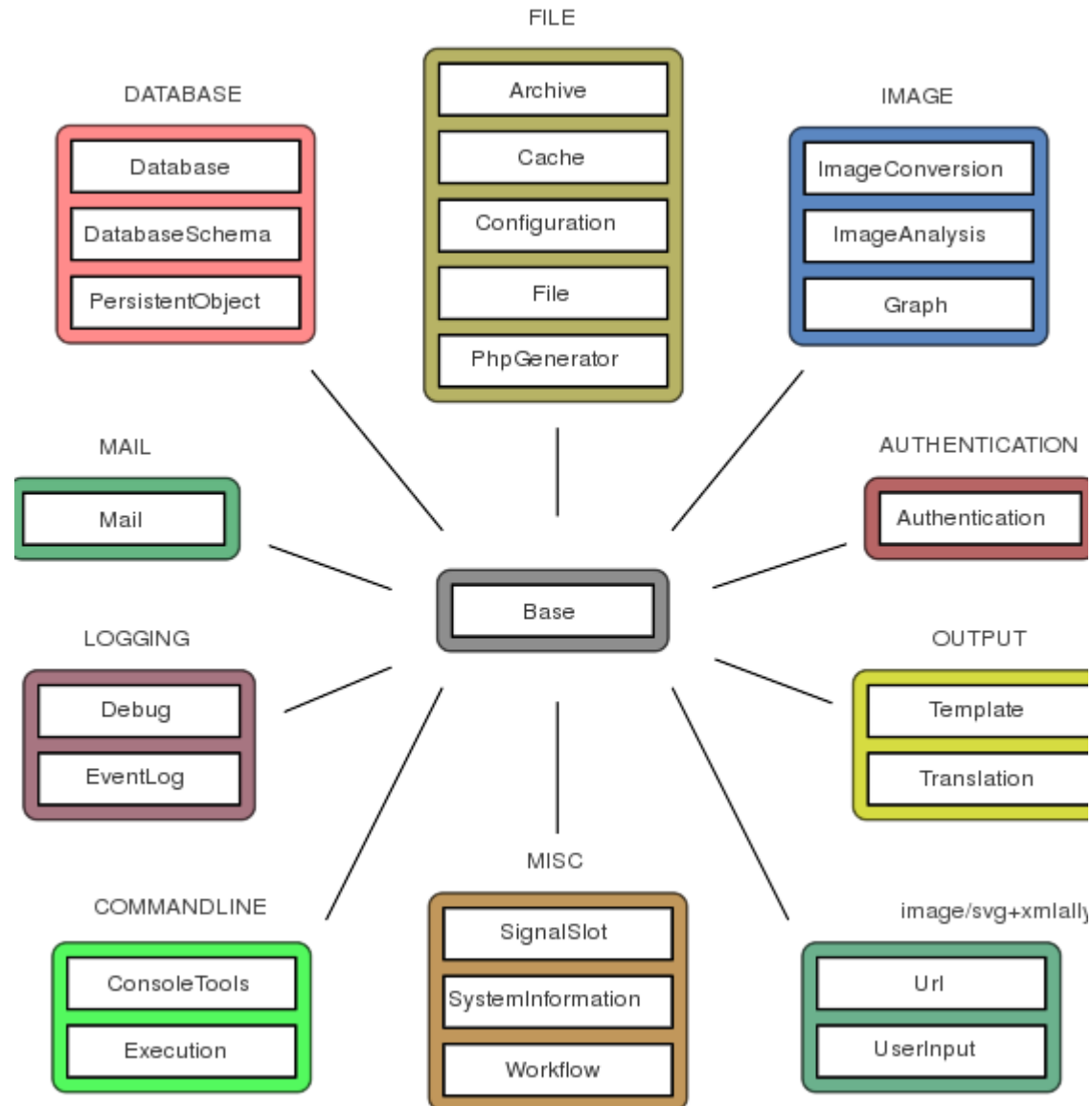
- Clashes in first part of the classname
- Needs installation into correct place for development

Versions for components have three elements: x.y.z, which have the following meaning:

- x: major version number. A component with major version 0 can never be released publically (beta). It will only increase when there is a backwards compatible break in the component's API.
- y: minor version number, is used for all feature additions.
- z: mini version number, is used to denote bugfixes only. This third part can also be a string in the set: (alpha, beta1, beta2, betaN, rc1, rc2, rcN).

x and y show the version number of the component, the z is an addition showing the state (beta etc) or which bugfix release it is.

# Overview



### Installation:

```
// download the bundle from http://ez.no/download/ez_components
tar -xjf ezcomponents-2007.1beta1.tar.bz2
pwd
```

### Setup:

```
<?php
ini_set( 'include_path', '/home/httpd/ezcomponents-2007.1beta1..' );
require 'Base/src/base.php';

function __autoload( $className )
{
    ezcBase::autoload( $className );
}
?>
```

### Installation:

```
pear channel-discover components.ez.no
pear remote-list -c ezc
pear install ezc/ezcomponents
```

### Setup:

```
<?php
require 'ezc/Base/base.php';

function __autoload( $className )
{
    ezcBase::autoload( $className );
}
?>
```

## Supported Handlers:

- MS SQL
- MySQL
- PostgreSQL
- Oracle
- SQLite

## Using Instances:

```
<?php
require 'ezc-setup.php';

$db1 = ezcdbFactory::create( 'mysql://root@localhost/ezc' );
ezcdbInstance::set( $db1, 'ezc' );

$strunk = ezcdbFactory::create( 'mysql://root@localhost/trip' );
ezcdbInstance::set( $strunk, 'ezt' );

$db = ezcdbInstance::get( 'ezt' );
echo $db->getName();
?>
```

Allows writing SQL that works for all supported databases, without having to deal with differences in SQL dialects.

Supported SQL categories:

- SELECT: `$db->createSelectQuery();`
- INSERT: `$db->createInsertQuery();`
- UPDATE: `$db->createUpdateQuery();`
- DELETE: `$db->createDeleteQuery();`

Query object:

- Uses a "fluent-interface"
- Exposes an expression object through `$db->e`.

Works by returning the object that a method is called on after the method does its work.

```
<?php
public function from()
{
    if ( $this->fromString == '' )
    {
        $this->fromString = 'FROM ';
    }
    // ...
    $this->fromString .= join( ', ', $tables );
    return $this;
}
?>
```

Allows for chaining of calls:

```
<?php
$q->select( '*' )->from( 'quotes' )
->where( $e->eq( 'author', $q->bindValue( 'Robert Foster' ) ) );
?>
```

- Safer: No manual quoting required
- Faster: The query is only compiled once

```
<?php
$country = $normalized_name = $name = $province = $lat = $lon = null;
$sq = $db->createInsertQuery();
$sq->insertInto( 'city' )
->set( 'country', $sq->bindParam( $country ) )
->set( 'normalized_name', $sq->bindParam( $normalized_name ) )
->set( 'name', $sq->bindParam( $name ) )
->set( 'province', $sq->bindParam( $province ) )
->set( 'lat', $sq->bindParam( $lat ) )
->set( 'lon', $sq->bindParam( $lon ) );
$stmt = $sq->prepare();

do {
    $line = fgets( $fd ); if ( feof( $fd ) ) break;

    $elements = explode( "\t", $line );
    list( $country, $normalized_name, $name, $province, $lat, $lon ) = $elements;
    $stmt->execute();
} while ( true );
?>
```

- Sits on top of Database
- Object Relational Model
- Based on Hibernate - not Active Record
- Uses definition files, which can be auto generated by a script as part of the PersistentObjectDatabaseSchemaTiein:

```
php rungenerator.php  
-s mysql://root@localhost/geolocation -f mysql /tmp
```

- The classes itself can also be generated by this script
- Work is being done to allow definitions to be generated on the fly from annotated source code

# PersistentObject Example

```
<?php
require 'ezc-setup.php';
$session = new ezcPersistentSession(
    ezcDbInstance::get(), new ezcPersistentCodeManager( "path/to/definitions" )
);

// Creating New Objects
$object = new City();
$object->normalized_name = "dieren";
$object->name = 'Dieren';
$object->country = 'NL';
$session->save( $object );

// Finding Objects
$q = $session->createFindQuery( 'City' );
$q->where(
    $sq->expr->like( 'name', $sq->bindValue( 'oslo%' ) )
)
->orderBy( 'country', 'name' )->limit( 10 );
$objects = $session->findIterator( $q, 'City' );

foreach ( $objects as $object )
{
}
?>
```

- Filters and Transformations
- Different backends: GD and ImageMagick

Supported filters:

- Colorspace: monochrome, sepia and grey
- Effects: border, noise, swirl
- Geometry: crop, scale
- Thumbnail
- Watermark

# ImageConversion

## Album thumbnailing

```
<?php
require 'ezc-setup.php';

// Setup
$filters = array();
$settings = new ezcImageConverterSettings(
    array( new ezcImageHandlerSettings( 'GD', 'ezcImageGdHandler' ) )
);
$converter = new ezcImageConverter( $settings );

// Create transformation
$filters[] = new ezcImageFilter(
    'croppedThumbnail', array( 'width' => 128, 'height' => 128 )
);
$converter->createTransformation(
    'doThumbnail', $filters, array( 'image/png' )
);

foreach ( glob ( '*..[Jj][Pp][Gg]' ) as $file )
{
    $converter->transform(
        'doThumbnail', $file, "thumbnails/{$file}.png"
    );
}
?>
```

demo

```
<?php
require 'ezc-setup.php';
$mail = new ezcMailComposer();

// from and to addresses, and subject
$mail->from = new ezcMailAddress( 'john@doe.com', 'John Doe' );
$mail->addTo( new ezcMailAddress( 'dr@ez.no', 'Derick Rethans' ) );
$mail->subject = "Example of an HTML email with attachments";

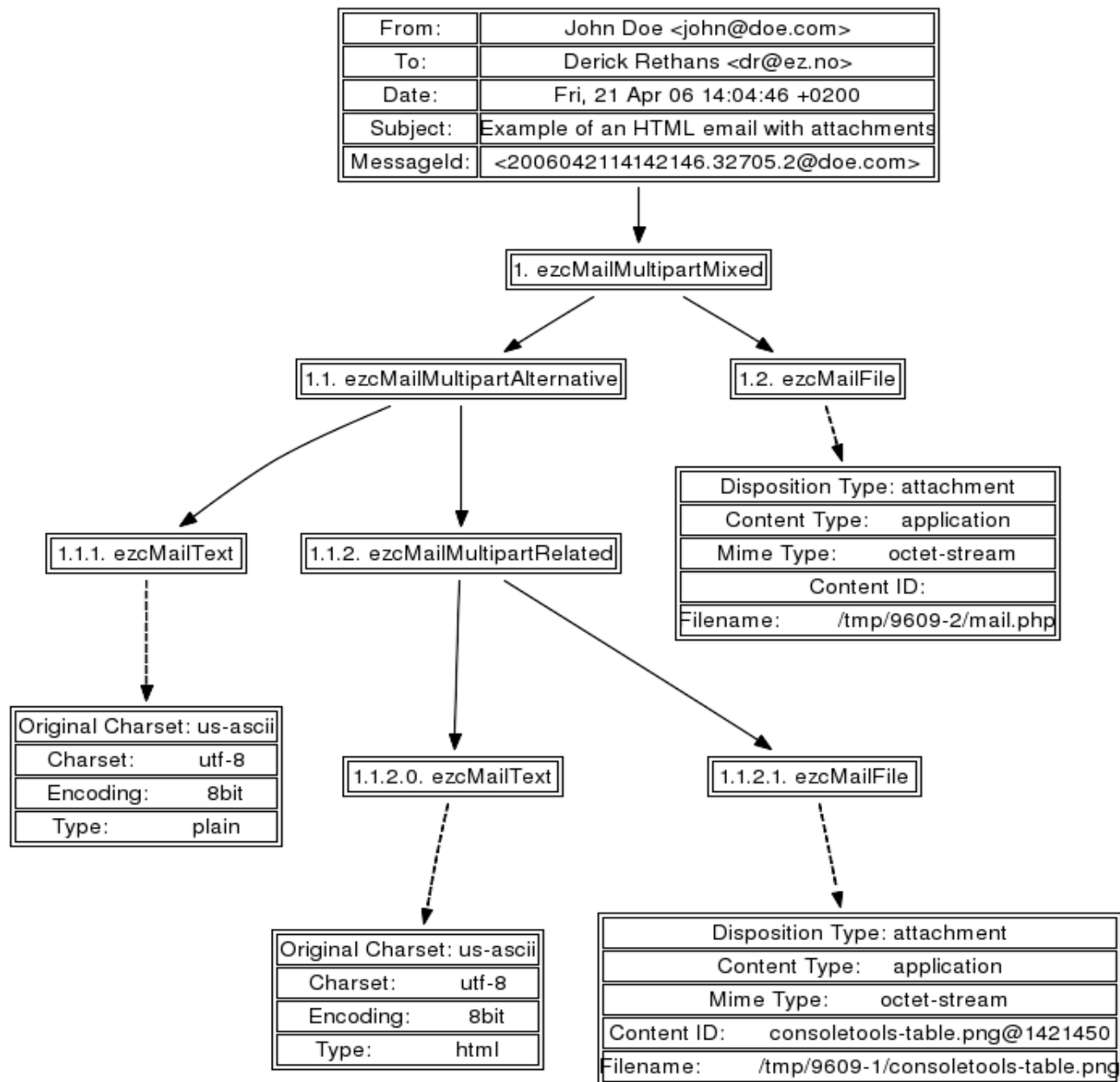
// body: plain
$mail->plainText = "Here is the text version of the mail.";

// body: html
$mail->htmlText = <<<ENDHTML
<html>Here is the HTML version of your mail with an image: <img
src='file://$dir/consoletools-table.png' /></html>
ENDHTML;

// add an attachment
$mail->addAttachment( "$dir/mail.php" );

// send the mail
$transport = new ezcMailTransportSmtplib( 'localhost', null, null, 2525 );
$transport->send( $mail );
?>
```

# Mail Creation Diagram



# Mail

## Retrieving and Parsing Mail

```
<pre><font size="4"><?php
require 'ezc-setup.php';
require 'mail-parse2b.php';

$set = new ezcMailFileSet( array( dirname( __FILE__ ) . "/mail-example.mail" ) );
$parser = new ezcMailParser();
$mail = $parser->parseMail( $set );
echo htmlspecialchars( formatMail( $mail[0] ) );

?>
```

Sending with:

- SMTP
- "MTA" (PHP's mail())

Reading from:

- POP3, IMAP, MBOX, Files, Variables

Supported formats:

- multipart/alternative: text/html mail
- multipart/mixed: attachments
- multipart/related: HTML mail with inline images
- multipart/digest: mailinglist digests

Overview:

Creates different kinds of diagrams:

- Line charts
- Bar charts
- Pie charts
- Radar charts

Drivers: GD, SVG, Flash

Renderers: 2D, 3D

Datasets: Array, Numeric, PDO, Average (Polynomial)

Axis: Date, Numeric, labeled, Logarithmic

Palettes: Black, eZ, Tango, User Defined

```
<?php
require 'ezc-setup.php';
header( 'Content-Type: image/svg+xml' );
list( $domains, $ips ) = require 'data-graph1.php';

$chart = new ezcGraphLineChart();

$chart->title = 'PHP Usage Statistics';
$chart->palette = new ezcGraphPaletteTango();
$chart->options->fillLines = 230;

$chart->legend->title = "Legend";

$chart->xAxis->font->maxFontSize = 12;
$chart->yAxis->font->maxFontSize = 12;
$chart->title->font->maxFontSize = 20;

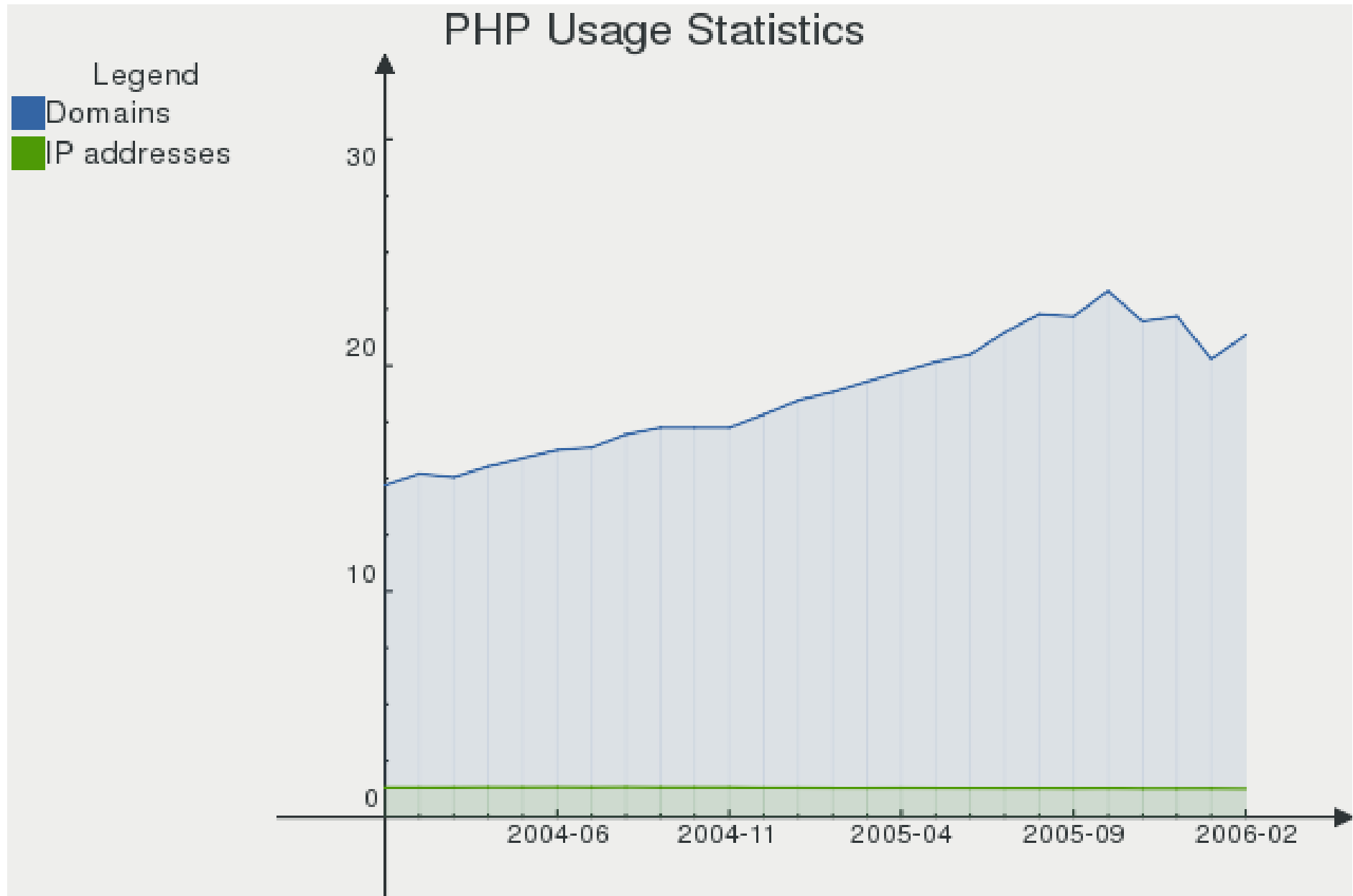
$chart->data['domains'] = new ezcGraphArrayDataSet( $domains );
$chart->data['domains']->label = 'Domains';
$chart->data['ips'] = new ezcGraphArrayDataSet( $ips );
$chart->data['ips']->label = 'IP addresses';

$chart->driver = new ezcGraphSvgDriver();
$chart->render( 600, 400, 'php://output' );

?>
```

# Graph

## Line-graph - Result



```
<?php
require 'ezc-setup.php';

$chart = new ezcGraphPieChart();
$chart->palette = new ezcGraphPaletteEzBlue();

$chart->title = 'Browser Partitioning';

$chart->data['browser'] = new ezcGraphArrayDataSet( array(
    'IE5' => 698,
    'IE6' => 17383,
    'IE7' => 708,
    'Firefox' => 10001,
    'Netscape' => 871,
    'Opera' => 584,
) );
$chart->data['browser']->highlight['Firefox'] = true;

$chart->driver = new ezcGraphSvgDriver();

$chart->renderer = new ezcGraphRenderer3d();
$chart->renderer->options->pieChartShadowSize = 12;
$chart->renderer->options->pieChartGleam = .5;
$chart->renderer->options->dataBorder = false;
$chart->renderer->options->pieChartHeight = 16;
$chart->renderer->options->legendSymbolGleam = .5;

$chart->renderer->options->pieChartOffset = 180;

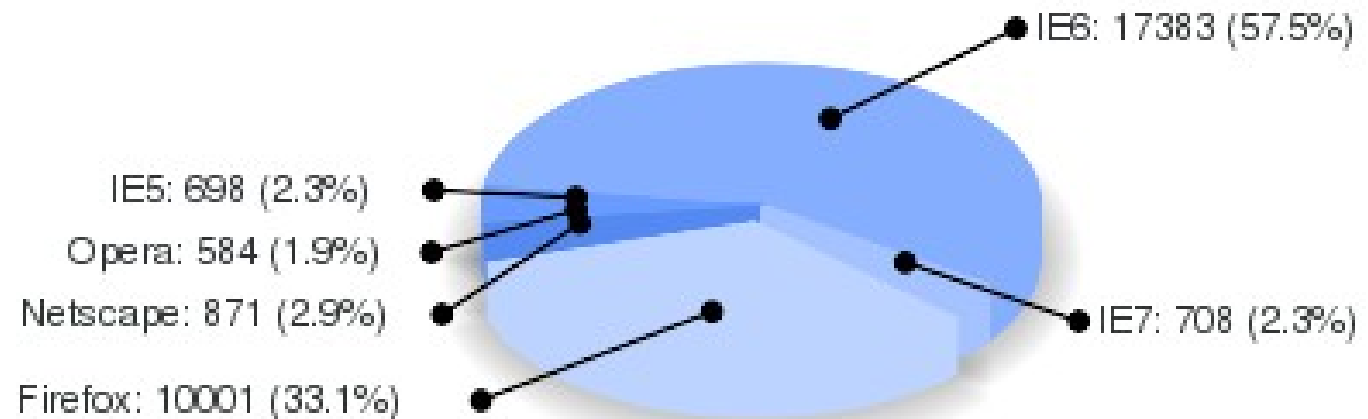
header( 'Content-Type: image/svg+xml' );
$chart->render( 560, 200, 'php://output' );
?>
```

The Graph component outputs:

- PNG/JPG, through the `ezcGraphGdDriver()`
- SVG, through the `ezcGraphSvgDriver()`
- Flash, through the `ezcGraphFlashDriver()`

### Browser Partitioning

- IE5
- IE6
- IE7
- Firefox
- Netscape
- Opera



demo

# What's Next?

## Additions for eZ Components 2007.1

### New components:

- Authentication: Authentication against database, Typekey, (OpenID)
- Workflow: Workflow engine

### Additions to components:

- Database: SQL Server driver
- Graph: Radar charts, PDO data sets
- Mail: Caching, SSL support
- PersistentObject: Class stub generation
- Template: Caching

Roadmap: <http://issues.ez.no/RoadMap.php?Id=487>

Questions anybody?

Resources:

- Download: <http://ez.no/ezcomponents/download>
- Documentation: <http://ez.no/doc/components/overview>
- Mailinglist: <http://lists.ez.no/mailman/listinfo/components>
- IRC: #ezcomponents @ Freenode
- These Slides: <http://files.derickrethans.nl/ezc-php-norge.pdf>