



# **ABOUT PHP**

Bun Wong

PHP 是什麼？

# PHP 的開發歷史



Rasmus Lerdorf

PHP 原本的簡稱為 Personal Home Page，是 Rasmus Lerdorf (雷斯馬) 為了要維護個人網頁，而用 C 語言開發的一些 CGI 工具程式集，來取代原先使用的 Perl 程式。

最初這些工具程式用來顯示 Rasmus Lerdorf 的個人簡歷，以及統計網頁流量。他將這些程序和一些表單解析器整合起來，稱為 PHP/FI。PHP/FI 能和數據庫連接，產生簡單的動態網頁程式。

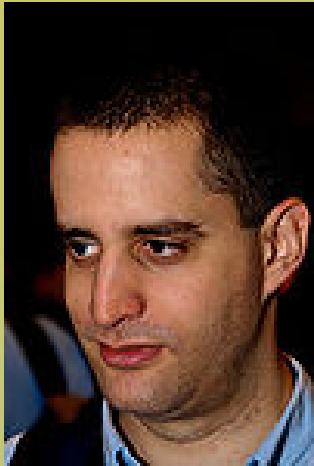
# PHP 的開發歷史



Rasmus Lerdorf

Rasmus Lerdorf 在 1995 年 6 月 8 日將 PHP/FI 開源，希望可以透過社區來加速程序開發與尋找錯誤。這個發布的版本命名為 PHP 2，已經有今日 PHP 的一些雛型，比如有類似 Perl 的變量命名方式、表單處理功能、以及嵌入到 HTML 中執行能力。程式語法上也類似 Perl，不過更簡單、更有彈性，但功能相當有限。

# PHP 的開發歷史



Zeev Suraski



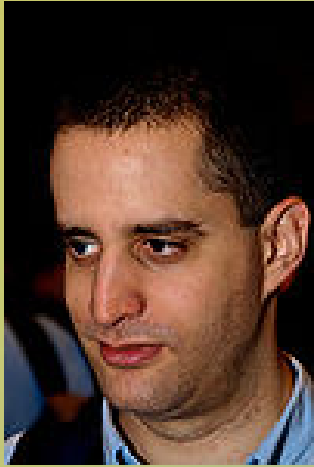
Andi Gutmans

在 1997 年，任職於 Technion IIT 公司的兩個以色列程序設計師：[Zeev Suraski](#) 和 [Andi Gutmans](#)，重寫了 PHP 的解析器，成為 PHP 3 的基礎，而 PHP 也在這個時候改稱為 PHP :

Hypertext Preprocessor。經過幾個月測試，開發團隊在 1997 年 11 月發布了 PHP/FI 2，隨後就開始 PHP 3 的開放測試，最後在 1998 年 6 月正式發布 PHP 3

。

# PHP 的開發歷史



Zeev Suraski

Zeev Suraski 和 Andi Gutmans 在 PHP 3 發布後開始改寫 PHP 的核心，這個在 1999 年發布的解析器稱為 Zend Engine，他們也在以色列的 Ramat Gan 成立了 Zend Technologies 來管理 PHP 的開發。



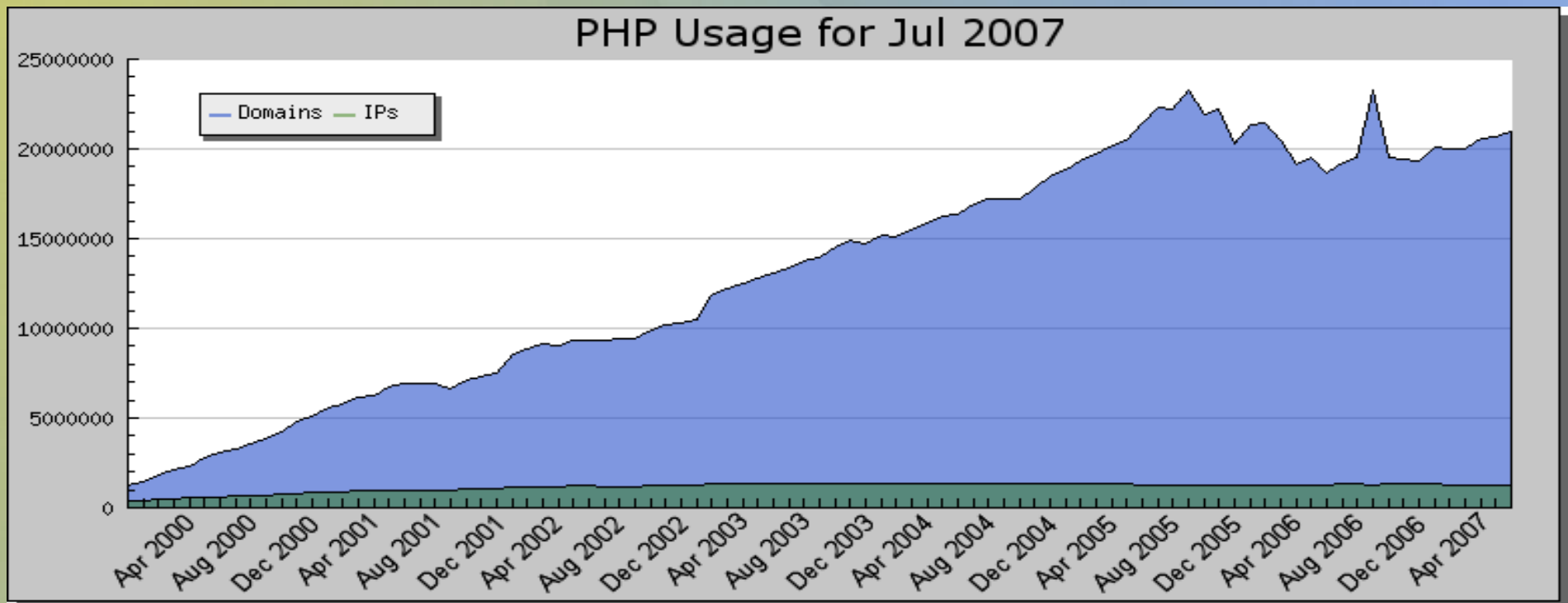
Andi Gutmans

在 2000 年 5 月 22 日，以 Zend Engine 1 為基礎的 PHP 4 正式發布，2004 年 7 月 13 日發布了使用了 Zend Engine 2 的 PHP 5。

# PHP 的應用

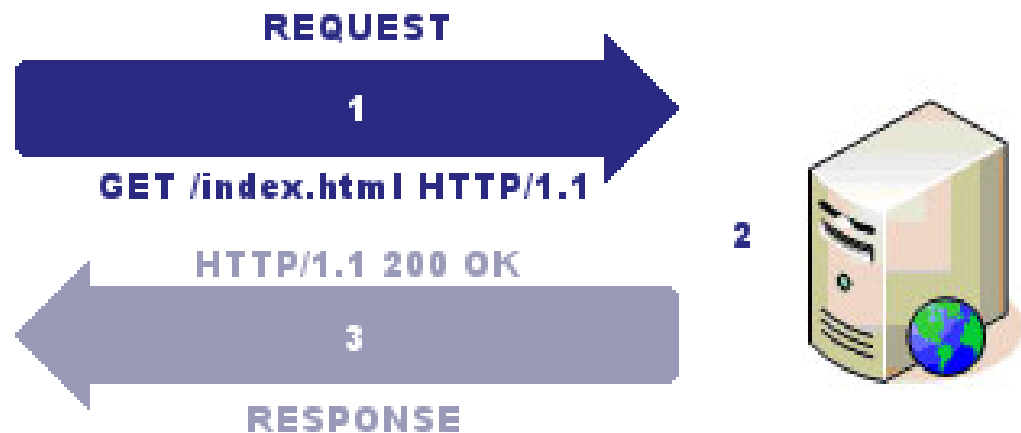
PHP 作為 LAMP 的其中一部分，最初就是設計成 Web 服務器端腳本語言，這也是 PHP 應用最廣的部份。

PHP 目前已經是全世界最受歡迎的服務器端腳本語言，跨平台的特性更是讓 PHP 廣為流傳，目前世界上有超過 2000 萬台服務器安裝有 PHP。



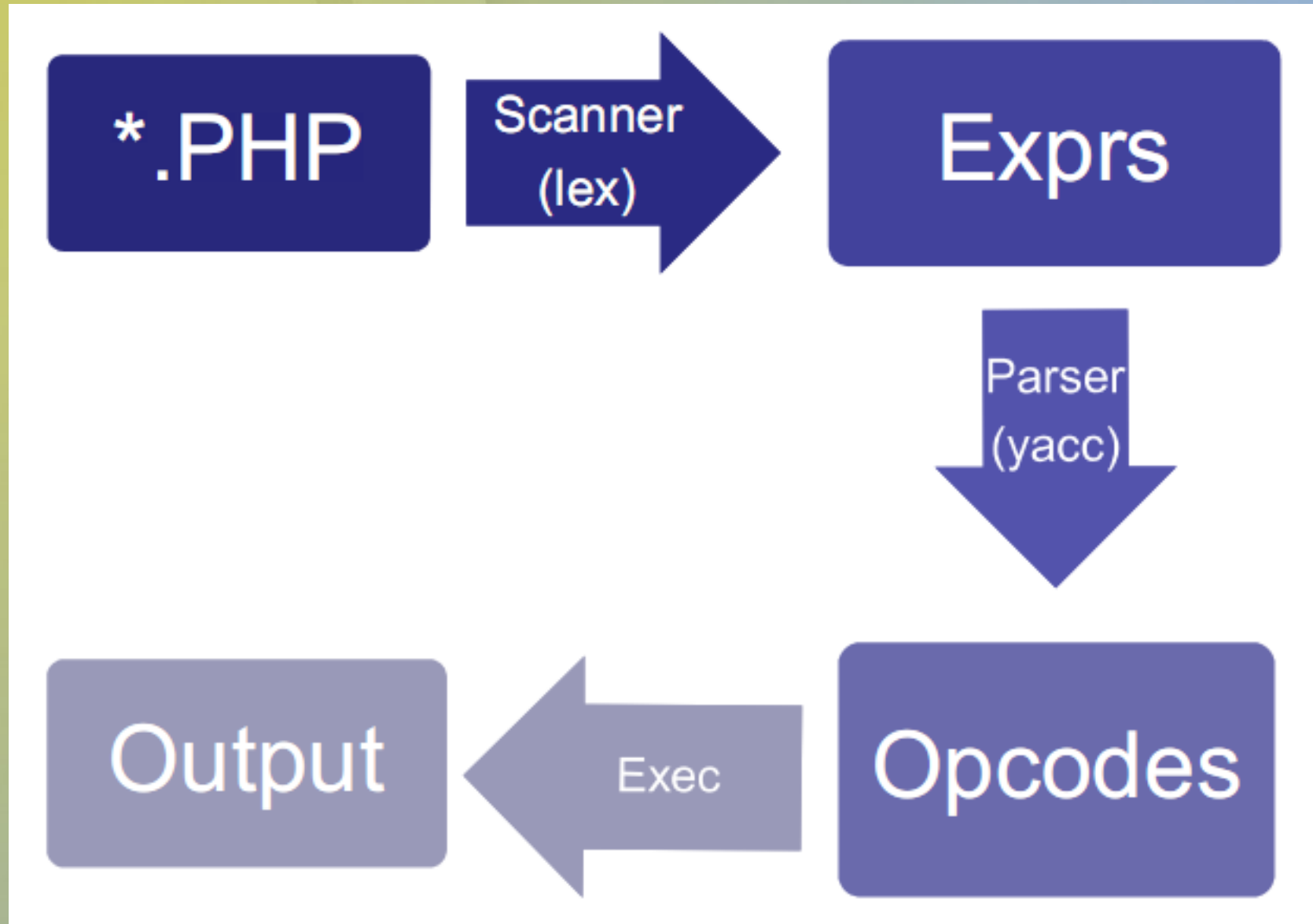
PHP 怎樣工作？

# HTTP Request-Response Cycle

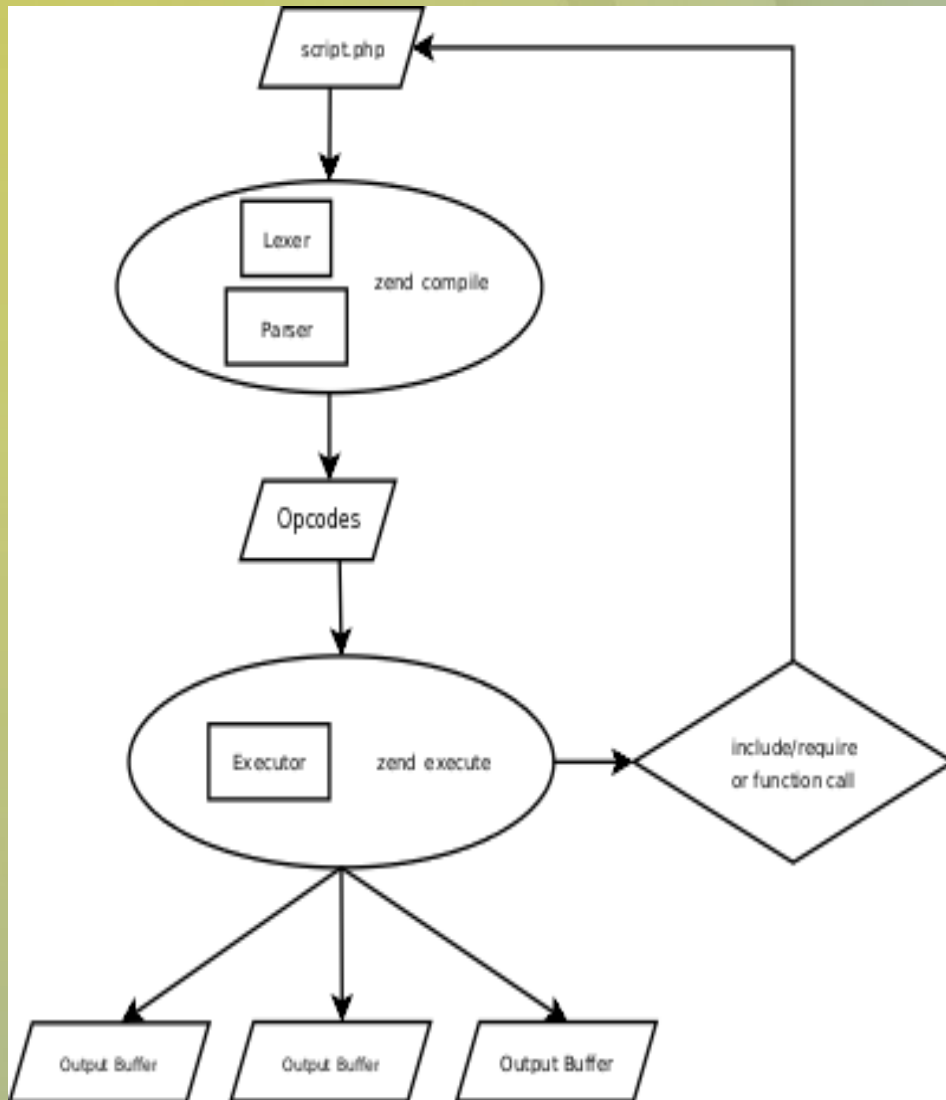


1. 客戶端請求
2. 服務器處理請求 (Apache, PHP) 并響應客戶端
3. 客戶端解析響應 (HTML, XML, JSON etc...)

# PHP Parse/Execute Process



# PHP Parse/Execute Process



1. Scanning (Lexing) , 將 PHP 代碼轉換成語言片段 (Tokens)

2. Parsing , 將 Tokens 轉換成簡單而有意義的表達式

3. Compilation , 將表達式編譯成中間語言 (OPcodes)

4. Execution , 順次逐行地執行 Opcodes , 從而實現 PHP 腳本的功能

# Zend Engine 和 OPcode

更確切的來說，以上的工作都是通過的 PHP 的核心 **Zend Engine (ZE)** 來實現的。

ZE 是一個虛擬機，正是由於它的存在，所以才能使得我們寫的 PHP 腳本，完全不需要考慮所在的操作系統類型是什麼。

**OPcode** 是一種 PHP 的腳本經過 ZE 編譯後的中間語言，就像的 Java 的 ByteCode 或者 .NET 中的 CIL。

Ze 支持超過 130 條指令 (OPcodes)，包括從最簡單的 ZEND\_ECHO (echo) 到複雜的 ZEND\_INCLUDE\_OR\_EVAL (include, require)，所有我們編寫的 PHP 的都會最終被處理為這 130 多條指令的序列，從而最終被執行。

# A sample (Scan)

PHP4.2 提供了一個函數 `token_get_all`，這個函數可以將一段 PHP 代碼 Scan 成 Tokens。

```
<?php
echo "Hello World";
$a = 1 + 1;
echo $a;
?>
```

```
array
 0 =>
  array
    0 => int 367
    1 => string '<?php'
' (length=7)
 2 => int 1
1 =>
  array
    0 => int 316
    1 => string 'echo' (length=
    2 => int 2
2 =>
  array
    0 => int 370
    1 => string ' ' (length=1)
    2 => int 2
3 =>
  array
    0 => int 315
    1 => string '"Hello World"'
    2 => int 2
4 => string ';' (length=1)
5 =>
  array
    0 => int 370
    1 => string '
' (length=2)
 2 => int 2
6 =>
  array
    0 => int 309
    1 => string '$a' (length=2)
    2 => int 3
7 =>
  array
    0 => int 370
    1 => string ' ' (length=1)
    2 => int 3
8 => string '=' (length=1)
9 =>
  array
    0 => int 370
    1 => string ' ' (length=1)
    2 => int 3
10 =>
  array
    0 => int 305
    1 => string '1' (length=1)
    2 => int 3
11 =>
  array
    0 => int 370
    1 => string ' ' (length=1)
    2 => int 3
12 => string '+' (length=1)
13 =>
  array
    0 => int 370
    1 => string ' ' (length=1)
    2 => int 3
14 =>
  array
    0 => int 305
    1 => string '1' (length=1)
    2 => int 3
15 => string ';' (length=1)
16 =>
  array
    0 => int 370
    1 => string '
'
```

# A sample (Parse & Compile)

分析返回結果可以發現，源碼中的字符串，字符，空格，都會原樣返回。每個源代碼中的字符，都會出現在相應的順序處。標籤，操作符，語句等，都會被轉換成一個包含 {TokenID, 源碼, 行號} 的數組，TokenID 跟 OPCODE 標識一一對應。

最終，腳本被解析成 OPCODE :

```
* ZEND_ECHO      'Hello World'  
* ZEND_ADD       ~0 1 1  
* ZEND_ASSIGN    !0 ~0  
* ZEND_ECHO      !0
```

并被 ZE 執行輸出至 Web 服務器 (Apache) 最終響應給客戶端。



**THE END ;)**