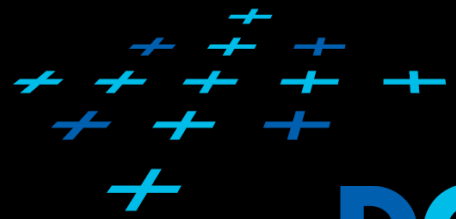


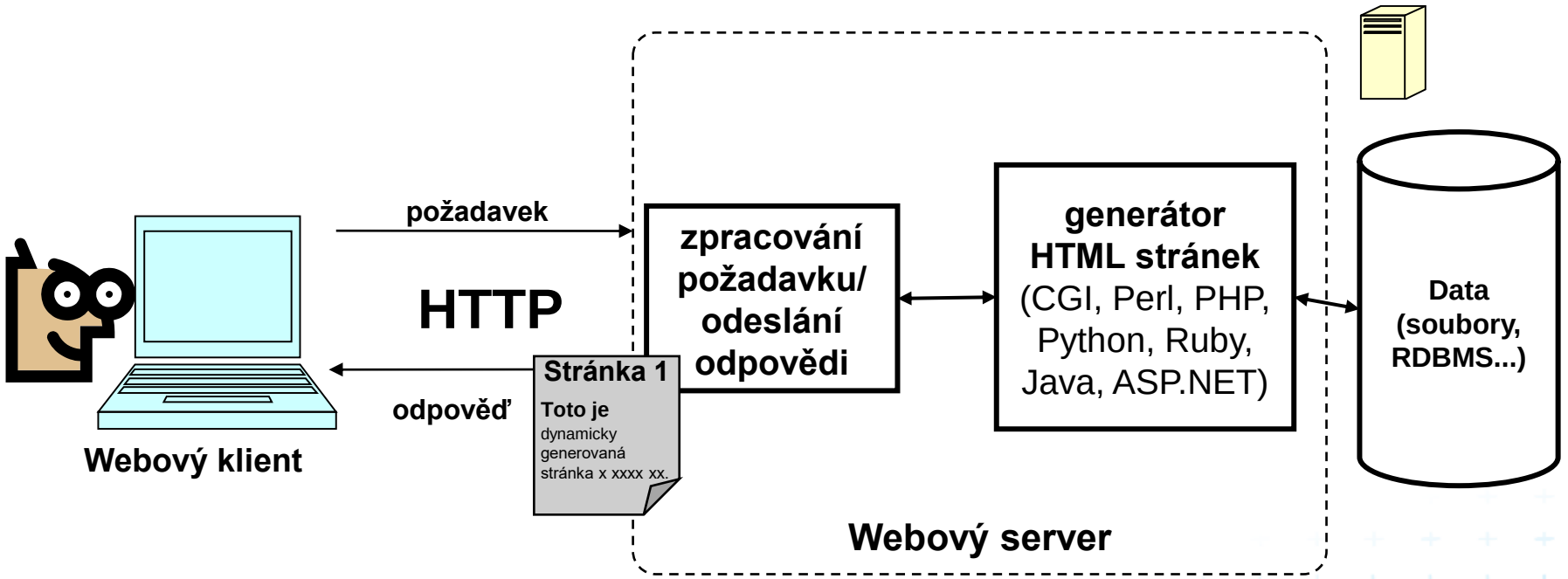
DCGI

KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE

Logika na straně server jazyk PHP

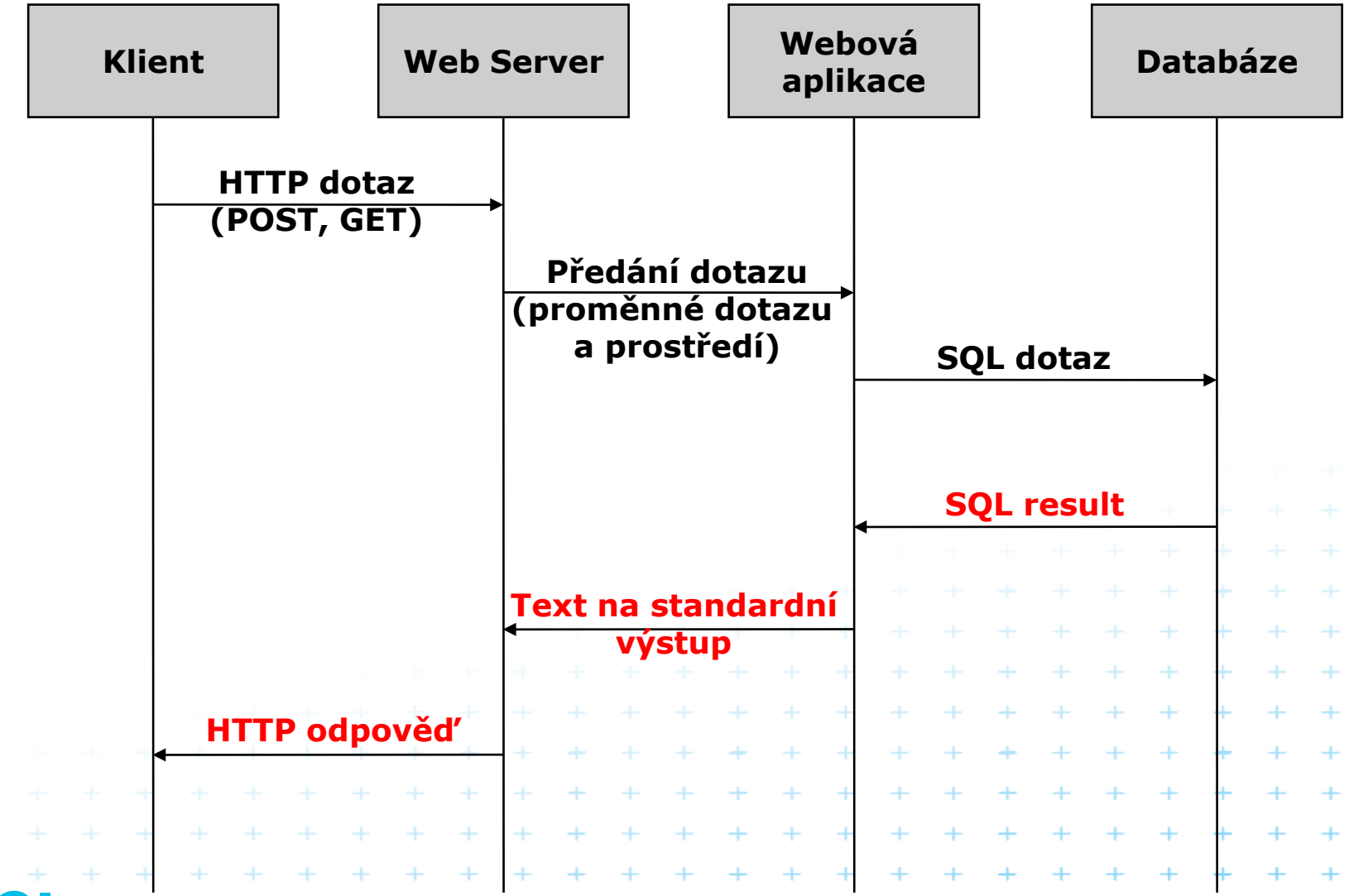
Martin Klíma

Dynamický web

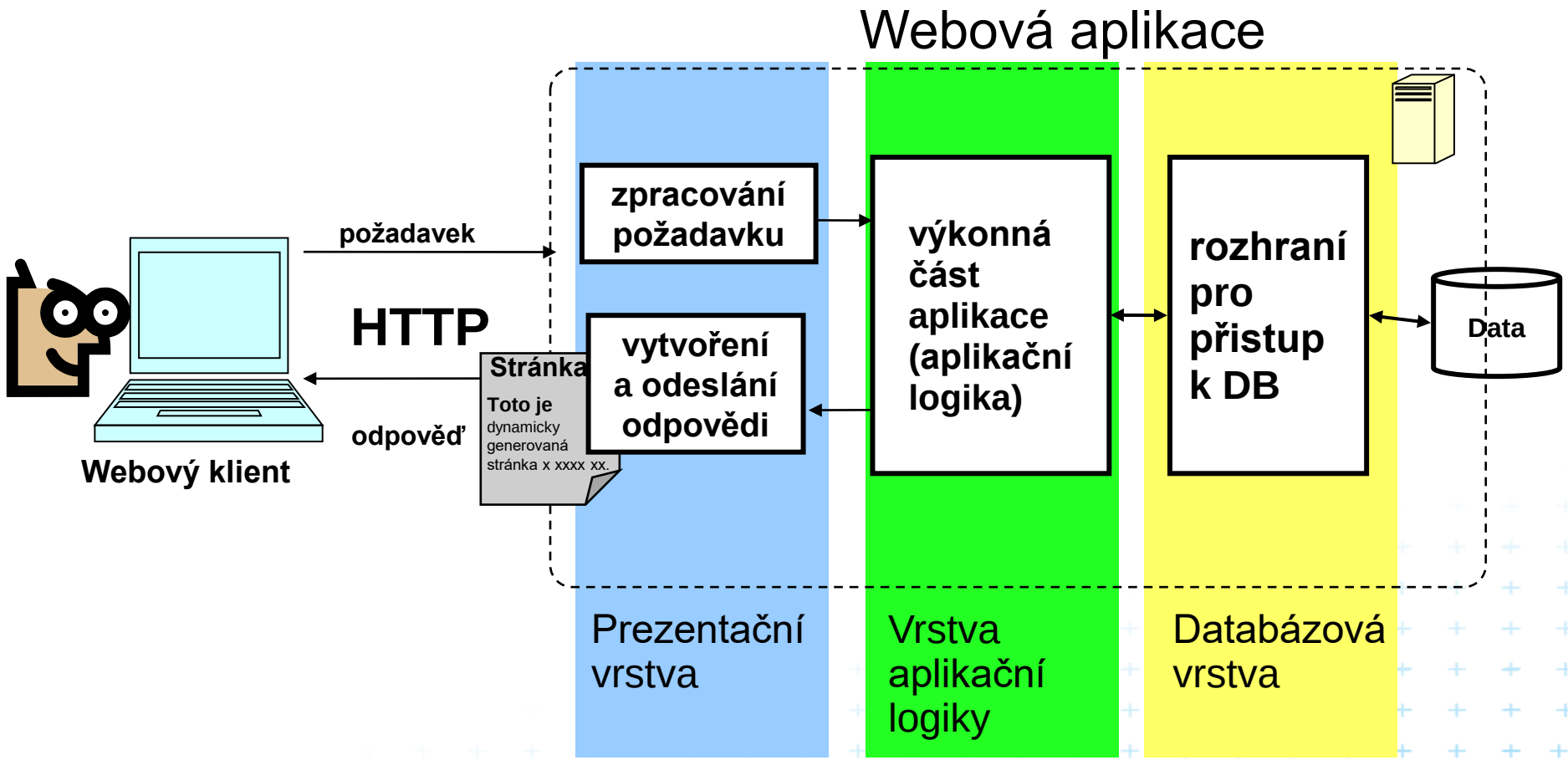


DCGI

Dynamický web – proces vzniku stránky



Architektura webové aplikace – 3 vrstvy

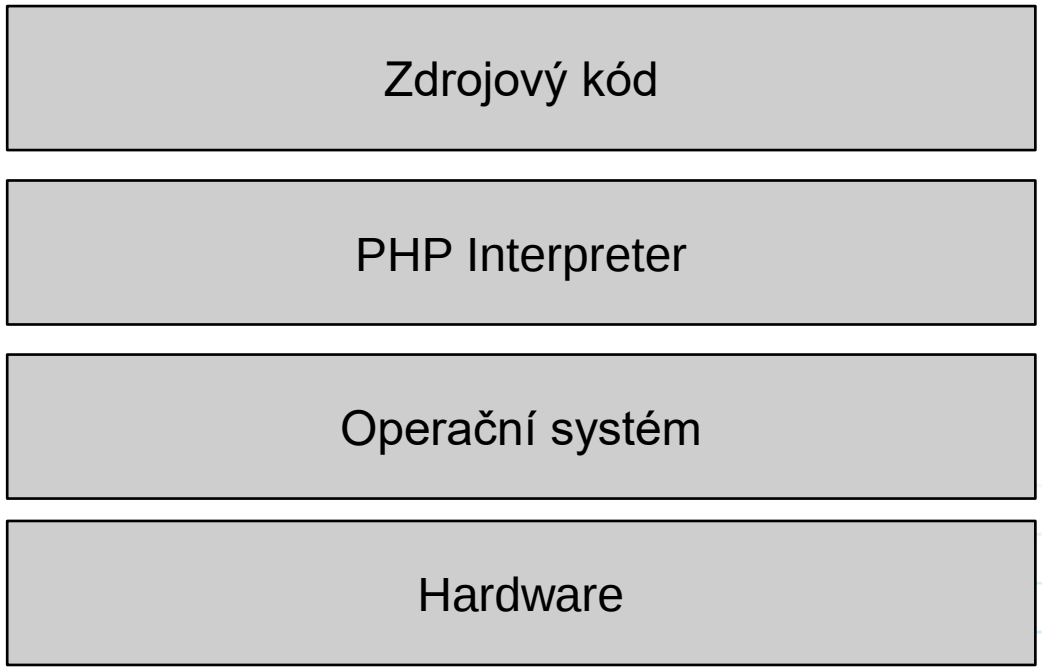


DCGI

Personal Home Pages | PHP Hypertext Preprocessor

PHP

PHP je platformově neutrální jazyk



DCGI

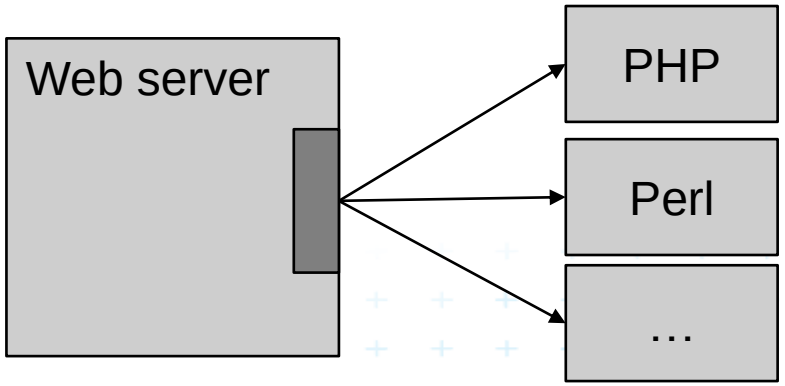
Jak se to slučuje s webem?

- Několik možností instalace
  - Jen PHP – to se nepoužívá

```
php -S localhost:8000
```

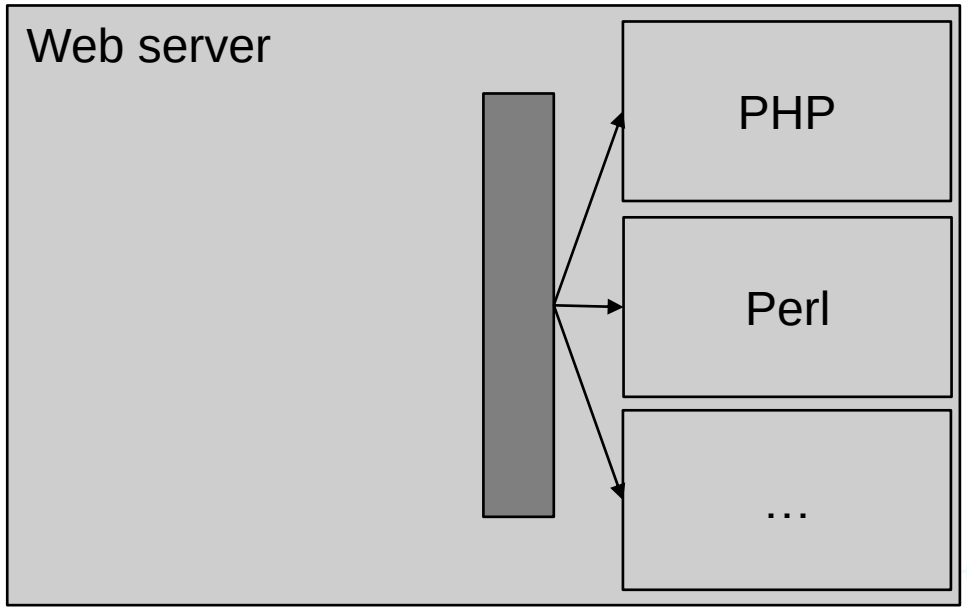
  - Jako externí modul přes CGI nebo FastCGI
    - CGI = Common Gateway Interface
    - Web server volá externí programy
    - Připraví prostředí
    - Zpátky dostane text
    - Server řeší http, exerní program řeší generování výstupu
    - Je to dosti neefektivní

      Proto existují různá další rozšíření, například FastCGI
    - Je to bezpečné



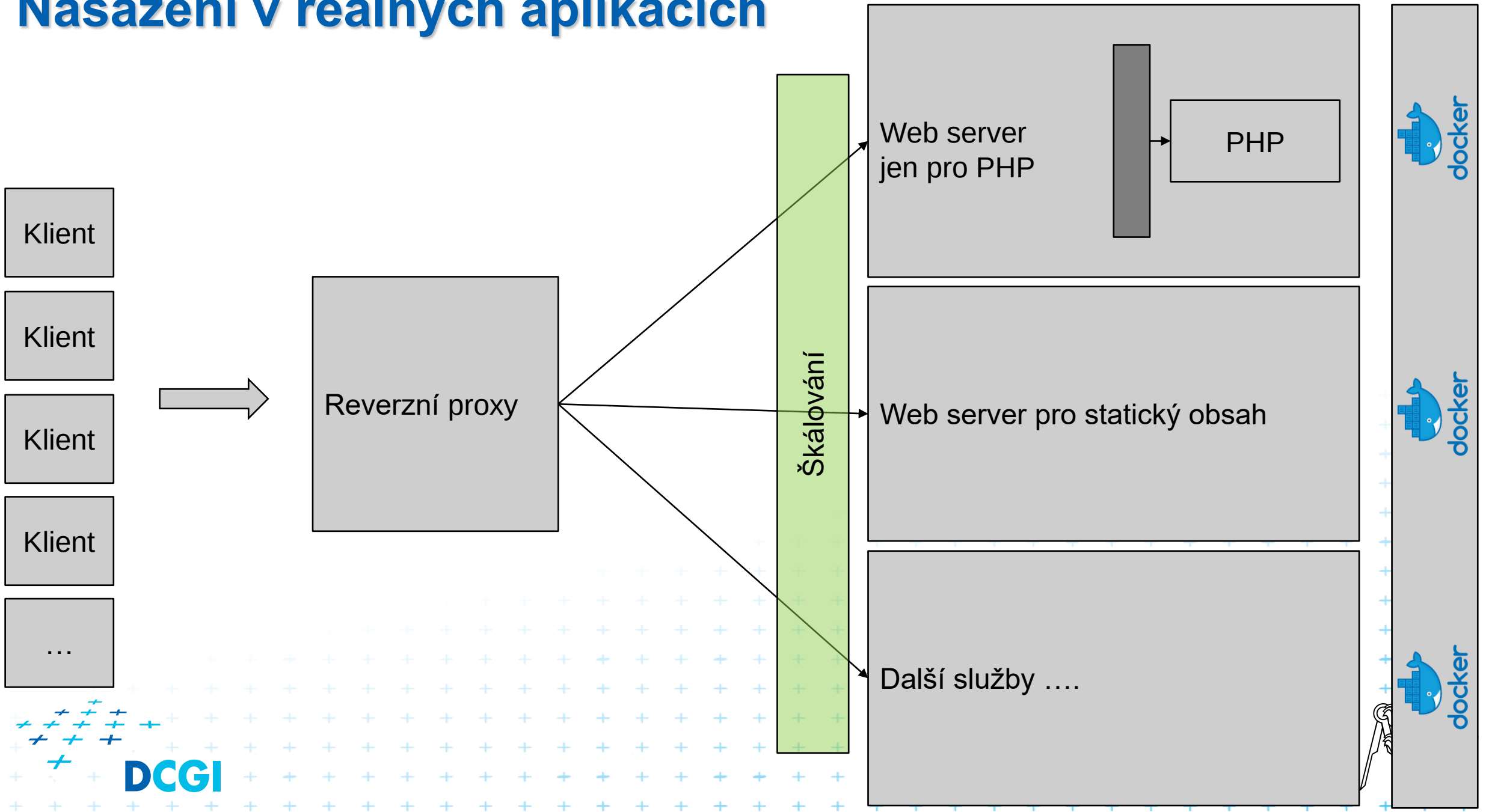
PHP a web server pokračování

- NSAPI, ISAPI
  - Vnitřní modul serveru
  - Společná paměť s webovým serverem
  - Není nutné vždy znovu spouštět nový modul
  - Méně bezpečné, ale rychlé



DCGI

Nasazení v reálných aplikacích



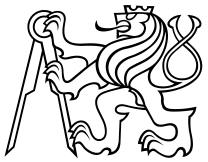
DCGI

Jak si nainstalovat PHP

- LAMP
  - Linux, Apache, Mysql, PHP
- WAMP
  - Windows, Apache, Mysql, PHP

Docela pěkný balíček XAMPP

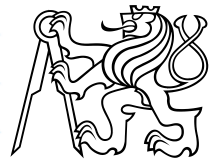
- Apache, MariaDB, PHP, Perl … a k tomu ještě pár věcí
- https://www.apachefriends.org/



Když už PHP máme

- Funguje jako klasický interpreter
  - režim interaktivní php -a
  - režim dávkový php *jmeno_souboru*

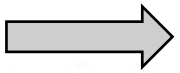
```
php > for ($i=1; $i<=10; $i++) {
php { echo "{$i} Ahoj\n";
php { }
1 Ahoj
2 Ahoj
3 Ahoj
4 Ahoj
5 Ahoj
6 Ahoj
7 Ahoj
8 Ahoj
9 Ahoj
10 Ahoj
php >
```



Dávkový režim

- Vstupem je zdrojový soubor
- Neoznačené bloky jsou předány přímo na výstup
- Označené bloky pomocí <?php *blok* ?> je interpretován jako PHP program

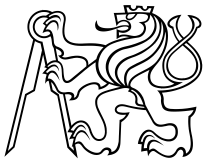
```
<!doctype php>
<html>
    <head>
        <title>Můj první pokus v PHP</title>
    </head>
    <body>
        <h1><?php
                echo "Ahoj světe";
        ?></h1>
    </body>
</html>
```



```
<!doctype php>
<html>
    <head>
        <title>Můj první pokus v PHP</title>
    </head>
    <body>
        <h1>Ahoj světe</h1>
    </body>
</html>
```

PHP (PHP Hypertext Preprocessor)

- Procedurální jazyk
- C-like syntax
- Množství funkcí v integrovaných knihovnách i modulech
- Jednoduchá integrace do webových serverů
- Od verze PHP5 solidní podpora OOP
- Není výhradně objektový



Historie PHP

Počáteční autor: Rasmus Lerdorf

Další důležití autoři: Zeev Surashi a Andi Gutmans

- 1995: PHP 1.0 - Personal Home Page Tools (PHP Tools)
- 2001: PHP 4.1.0 – všechny základní rysy dneška
- 2004: PHP 5.0.0 – nový objektový model
- 2008: PHP 5.3 – bezpečnostní vylepšení, jmenné prostory,…
- 2014: PHP 7 – mnoho vylepšení, jmenné prostory
- dnes PHP 8.x.x

PHP

Výhody

- Jednoduchý na učení
- Podpora pro webové funkce přímo v jazyce
- Multiplatformní
- Open Source
- Web hosting - masová podpora
- Velká komunita vývojářů

Nevýhody

- Netypovost, nekompiluje se, rychlost, ~~neobjektovost~~
- Někdy těžkopádný
- Některé věci v jazyce jsou nedotažené
- Svádí k mizernému způsobu programování

C-like syntax

- Příkazy oddělené středníkem ;
- Bloky sdružené pomocí { … }
- Komentáře:
  - // řádkové
  - /* blokové */
- Přiřazení pomocí ‘=’ $a=6
- Porovnání < , >, ==, !=
- Proměnné jsou uvedeny pomocí znaku $ - $promenna
- Kontrolní struktury: if (cond) {..} else {..}, while (cond) {..} , switch/case, for (startcond; increment; endcond) { }, foreach ($pole as $prvek),
- Prvky pole přístupné pomocí [ ] : $x[4] je 5. element pole $x
- Asociativní pole: $asocPole[“nazev_prvku”] = 'hodnota prvku';
- Funkce volané jménem s argumenty v pevném pořadí uzavřenými do ( ): substr (“retezec”,0,2)
- Case sensitive - $promenna není to samé jako $Promenna

DCGI

Ukázka syntaxe

```
$pole = array("Martin","Martin","Tomas","Radek","Petr");
// vypis
for ($i = 0; $i <= 2; $i++) {
  if (($i % 2) == 0) {
      echo (substr($pole[$i], 0, 1) . "<br />");
  } else {
      echo ('tohle je else');
  }
}
```

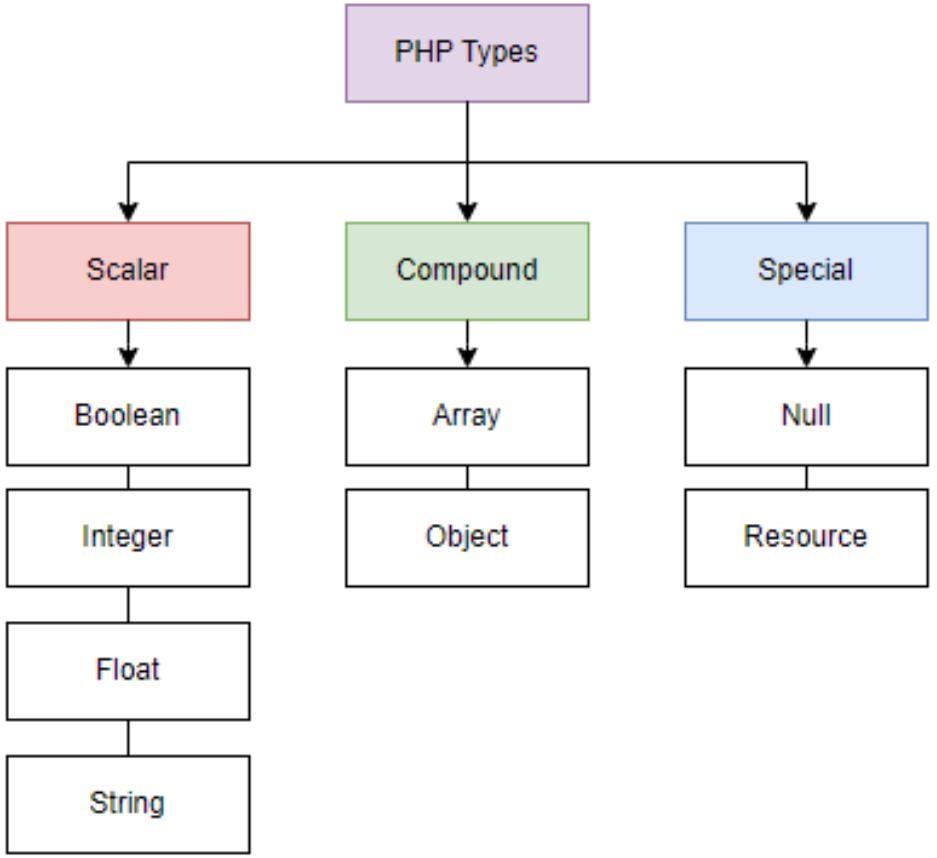
PHP proměnné, datové typy

Přetypování:

```
$x = (int) $y;
$x = (bool) $y;
$x = (float) $y;
$x = (double)$y;
$x = (string) $y;
$x = (array) $y;
$x = (object) $y;
```

Dotazování:

```
is_int($x);
is_bool($x);
is_float($x);
is_double($x);
is_string($x);
is_array($x);
is_object($x);
is_resource($x);
```



zdroj: https://www.phptutorial.net/php-tutorial/php-data-types/

Resource ukazuje na externí data jako je stream, databáze, soubor, apod.

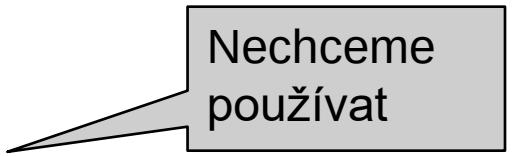
DCGI

PHP proměnné, datový typ

```php
<?php

   $x = "vyska";
   $$x = 10;

   echo $vyska; // 10
   echo "<br/>";
   echo $$x; //10

   $y = 1; //int
   $y = 1.0; // float
   $y = "abc"; // string


?>
```

Proměnné - viditelnost

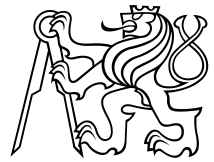
```
<?php
 $a = 1; $b = 2;
function Sum() {
    global $a, $b;
    $b = $a + $b;
 }

Sum();
echo $b;
?>
```



Proměnné viditelnost

- Superglobální proměnné
  - $_GET
  - $_POST
  - $_REQUEST
  - $_COOKIE
  - $_SESSION
  - $_ENV
  - $_FILES
  - $_SERVER
  - $GLOBALS

  - Tyto proměnné jsou viditelné vždy a všude. Není nutné na ně volat global



Proměnné pokračování

- Reference a hodnota

```php
<?php
	$jmeno = "František";

	echo "Jméno: ".$jmeno;

	//prirazeni hodnotou
	$jmeno2 = $jmeno;

	// reference na $jmeno
	$jmeno_ref = &$jmeno;

	//zmenime hodnotu promenne $jmeno
	$jmeno = "Hugo";

	echo "<br/>";

	//test
	echo "Jmeno: $jmeno, Jmeno2: $jmeno2, Jmeno_ref: $jmeno_ref";
?>
```

Zvláštnosti

- Datový typ string

Při použití " se obsah řetězce vyhodnotí a proměnná se nahradí hodnotou

Při použití ' se obsah řetězce nevyhodnocuje

```php
<?php

  $jmeno = "František";
  $prijmeni = "Vomáčka";

  echo "$jmeno, $prijmeni";
  echo "<br/>";
  echo '$jmeno, $prijmeni';
?>
```

DCGI

Konstanty

- Globální
  - patří ke jmennému prostoru
- Lokální
  - patří ke třídě, statický charakter

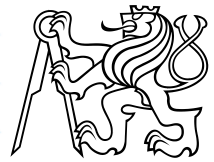
```php
<?php

// konstanty
class MojeTrida {
        public const MIN_VYSKA = 22;
}

const MAX_DELKA = 123.6;

echo MojeTrida::MIN_VYSKA;
echo MAX_DELKA;

?>
```

DCGI

Předdefinované proměnné

$_GET
$_POST
$_REQUEST
$_COOKIE
$_SESSION
$_ENV
$_FILES
$_SERVER
$GLOBALS

$php_errormsg
$HTTP_RAW_POST_DATA
$http_response_header
$argc
$argv



Předdefinované konstanty

…těch je mnoho, viz

http://www.php.net/manual/en/reserved.constants.php

Pole

- Pole je nejsilnější datový typ (hned po objektech)
- Všechna pole jsou asociativní

```
$pole = array();

$pole = array(4,5,6,7,8,9); // indexy od 0

$pole = array(5=>10, 20,30); // první index=5, další 6

$pole = array("pondělí"=>1, "úterý"=>2, "středa"=>3); // indexy jsou retezce

$pole[] = “xxx”; $pole[] = “yyy”;
```

Pole

- Vícerozměrná pole

```
$pole[1][30] = 20;
$pole[3][10] = 22;
```

- Iterování polí

```
foreach ($pole as $klic=>$hodnota) {
    echo "$klic = $hodnota <br>";
 }
foreach ($pole as $hodnota) {
    echo "hodnota pole: $hodnota <br>";
 }
```

Dotazování, mazání

```
if (isset ($pole[1][5]))
    echo "pole[1][5] je nastaveno";
 else echo "pole[1][5] neni nastaveno";

unset ($pole[1][5]);
```

Operátory

,
or
xor
and
print
= += -= *= /= .= %= &= |= ^= <<= >>=
? :
||
&&
|

^
&
== != === !==
< <= > >= <=>
<< >>
+ - .
* / % **
! ~ ++ -- (int) (float) (string) (array) (object) @
[
new

Řídící struktury

- if
- else
- elseif
- while
- do..while
- for
- foreach
- break
- continue
- switch
- declare
- return

If

```php
<?php

 $pole = array (1,2,5,6);

 if (sizeof($pole)>3)
  echo "velikost &gt; 3";
 elseif (sizeof($pole)>1)
  echo "velikost &gt; 1";
 else
  echo "pole je prázdne";

?>
```

DCGI

while, do - while

```php
$pole = array(1,2,4,5,6);
$i=0;
while($i < sizeof($pole)) {
    echo $pole[$i]."<br/>";
    $i++;
}

$i = 0;
do {
    echo $pole[$i]."<br/>";
    $i++;
} while ($i< sizeof($pole));
```

DCGI

For, foreach

```php
 $pole = array("pondělí"=>1, "úterý"=>2,
"středa"=>3);

 // selhava, protoze pole nema ciselne
indexy
 for ($i=0; $i<sizeof($pole); $i++) {
     echo $pole[$i]."<br/>";
 }

// univerzalni, v poradku
foreach ($pole as $klic=>$hodnota) {
     echo "index: $klic hodnota: $hodnota
<br/>";
 }
```

switch

```
 $jmeno = "Josef";
 switch ($jmeno) {
      case "Karel": echo "ahoj Karle";
break;
      case "Zdenek": echo "ahoj Zdenku";
break;
      case "Jarmila": echo "ahoj
Jarmilo";break;
      case "Josef":
      case "Pepa": echo "ahoj Pepo";
break;
      default: echo "ahoj neznámý";
 }
```

DCGI

Funkce a procedury

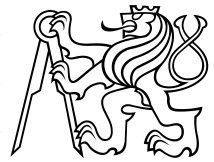
…jedno jsou

Funkce může a nemusí vracet hodnotu.

Návratový typ není deklarován.

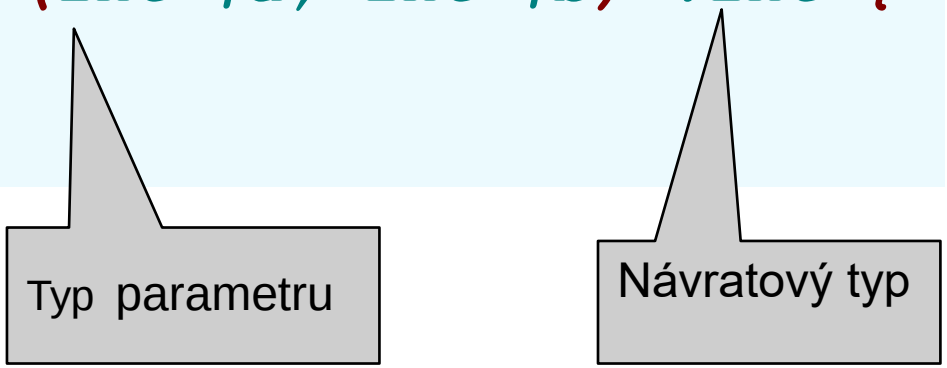
```php
function sectiPrvkyPole ($pole) {
 if (!is_array($pole)) return false;
 $vysledek = 0;
 foreach ($pole as $hodnota) {
   $vysledek += $hodnota;
 }
 return $hodnota;
}

$pole = array(1,2,3);

echo sectiPrvkyPole($pole);
```

DCGI



Funkce pork.

```php
function sectiDveHodnoty (int $a, int $b) :int {
 return $a + $b;
}
```



- Platí pro PHP 7 a vyšší.

DCGI

Zpracování chyb v PHP4 a 5

error_reporting()

set_error_handler()

```
error_reporting(E_ALL);

function my_error_handler ($severity, $msg,
$filename, $line_num) {
      // dostanu info o chybe a muzu si s ni
delat co chci
      echo "Závažnost: $severity <br/>Hláška:
$msg <br/> Soubor: $filename <br/> Číslo řádku:
$line_num <br/>";
}
set_error_handler("my_error_handler");
echo $xxx;
```

Výjimky v PHP5 a vyšší

- Je zde zaveden lepší způsob ošetřování výjimek.
- Podobnost s Javou.
- Jestliže je vygenerována výjimka (chyba), je vyroben nový objekt.
- Každá výjimka je rozšířením třídy Exception.
- Odvozením nové třídy lze vyrábět vlastní výjimky.

Výjimky PHP 5 a vyšší

```
class DevZeroException extends Exception {}
class NegativValueException extends Exception {}

function deleni ($a, $b) {
        try {  if ($b == 0) throw new DevZeroException();
               if ($a<0 || $b<0) throw new
NegativValueException();
               return $a/$b;
        }
        catch (Exception $e) {
               echo "doslo k nejake vyjimce!!!!";
               return false;
        }
//      catch (DevZeroException $e) { echo "nulou nelze delit";
//             return false;}
//      catch (NegativValueException $e2) {echo "negative value
odchyceno v ramci funkce"; return false;}
}
deleni(1,2);
deleni(1,0);
deleni(-1,5);
```

Vkládání souborů

- Použití knihoven = externí soubory
- Do místa direktivy vloží obsah referovaného souboru

- 2 základní způsoby
  - pouze jednou
  - iterativně

- Direktivy programu

```
include    "soubor"
include_once "soubor"

require "soubor"
require_once "soubor"
```

rozdíl mezi include a require:
funkční rozdíl není

include při nenalezení souboru pokračuje v běhu programu,
require končí kritickou chybou

DCGI

Dotazy?

DĚKUJI ZA POZORNOST