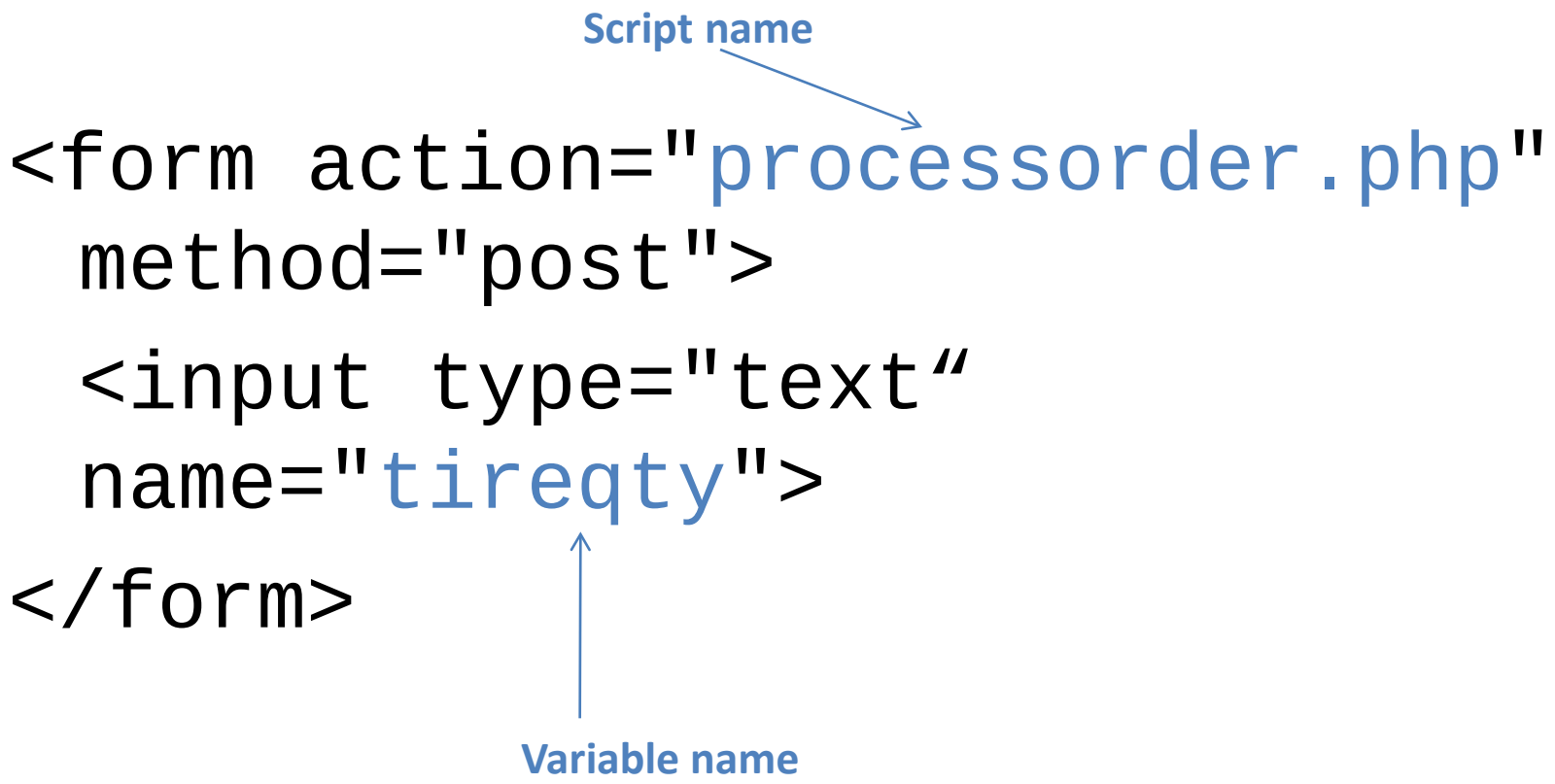
Introduction to Databases

PHP I

Key Concepts

- PHP in HTML
- Calling functions
- Form variables
- Identifies and data types
- Operators
- Decisions
- Conditionals
- Arrays
- Multi-dimensional arrays
- Sorting arrays
- Array manipulation

Calling a PHP Script



PHP Tag Styles

- XML style

```
<?php echo '<p>Order processed.</p>'; ?>
```

- Short style

```
<? echo '<p>Order processed.</p>'; ?>
```

- SCRIPT style

```
<script language='php'> echo '<p>Order
  processed.</p>'; </script>
```

Comments

- Multi-line comment

```
/* Author: Bob Smith
    Last modified: April 10
    This script processes orders
*/
```

- Single-line comment

```
//Start printing order
```

- Single-line shell-script style comment

```
#Start printing order
```

- Single-line comment ended by
  - End of line
  - End of php script tag

The date Function

- Outputs the current date
- Accepts a format string that specifies the date's format

Accessing Form Variables

- Short style
  `$tireqty`
  - Requires `register_globals` configuration
  - Security vulnerability
- Medium style
  `$_POST['tireqty']`
  - Default and recommended style
- Long style
  `$HTTP_POST_VARS['tireqty']`
  - Will be deprecated
  - Use for legacy server compatibility

String Concatenation

- Use period (.) as a string concatenation operator

```
echo $tireqty.' tires<br/>';
```

- Place string and simple variable inside double quotes

```
echo "$tireqty tires<br/>";
```

- Use a heredoc (<<<) for multiple lines

```
echo <<<theEnd
$tireqty tires
$sparkqty spark plugs
theEnd
```

Variables and Data Types

- Weakly typed
- Cast data type

```
$totalqty = 0;
$totalamount = (float)$totalqty;
```

- Change data type with `settype` function

```
bool settype(mixed var, string type);
settype($a, 'double');
```

- Determine data type with `gettype` function

```
string gettype(mixed var);
echo gettype($a);
```

Constants

- Use define function to create a constant
  ```
  define('TIREPRICE', 100);
  ```
- A constant name is not preceded by $
  ```
  echo TIREPRICE;
  ```
- Use `phpinfo();` to obtain a list of predefined PHP constants

Superglobals

Superglobal	Explanation
$GLOBALS	Array of global variables
$_SERVER	Array of server environment variables
$_GET	Array of variables passed via GET method
$_POST	Array of variables passed via POST method
$_COOKIE	Array of cookie variables
$_FILES	Array of variables related to file uploads
$_ENV	Array of environment variables
$_REQUEST	Array of all input except $_FILES
$_SESSION	Array of session variables

Operators

- Arithmetic operators
  `+, -, *, /, %`
- Assignment operator
  `=`
- Combined operators
  `+=, -=, /=, *=, %=, .=`
- Increment and decrement operators
  `--, ++`
- Reference operator
  `&`
- Comparison operators
  `==, ===, !=, !===, <>, <, >, <=, >=`
- Logical operators
  `!, &&, ||, and, or, xor`
- Bitwise operators
  `&, |, ~, ^, <<, >>`

Other Operators

- Instantiate a class
  `new`
- Access a class member
  `->`
- Type operator
  `instanceof`
- Ternary operator
  `?`
  ```
  ($grade >= 50 ? 'Passed' : 'Failed');
  ```
- Error suppression operator
  `@`
  ```
  $a = @(57/0);
  ```
- Command-line execution operator
  ``` `` ```
  ```
  $out = `ls –la`;
  echo '<pre>'.$out.'</pre>';
  ```
  ```

Variable Functions

- is_array
- is_double, is_float, is_real
- is_long, is_int, is_integer
- is_string
- is_bool
- is_object
- is_resource
- is_null
- is_scalar
- is_numeric
- is_callable

- isset
- unset
- empty
- intval
- floatval
- strval

Decisions

- Single-statement if

```
if($totalqty==0)
  echo 'You did not order anything<br/>';
```

- Code blocks

```
if($totalqty == 0) {
  echo '<p style="color:red">';
  echo 'You did not order anything</p>';
}
```

Using else

```
if ($totalqty==0)
  echo "You did not order anything<br/>";
else {
  echo $tireqty." tires<br/>";
  echo $sparkqty." sparkplugs<br/>";
}
```

Using elseif

```
if ($tireqty < 10){
  $discount = 0;
} elseif (($tireqty >= 10) && ($tireqty <=49)) {
  $discount = 5;
} elseif (($tireqty >=50) && ($tireqty <=99)) {
  $discount = 10;
} elseif ($tireqty > 100) {
  $discount = 15;
}
```

Switch Statement

```
switch($find) {
  case "a" :
     echo "<p>Regular customer</p>";
     break;
  case "b" :
     echo "<p>Customer referred by TV
  advertisement</p>";
     break;
  default:
     echo "<p>We do not know how the
  customer found us.</p>";
}
```

while and do..while

while loop

```
$num = 1;
while ($num <= 5) {
  echo $num."<br/>";
  $num++;
}
```

do…while loop

```
$num = 100;
do {
  echo $num."<br/>";
  $num--;
} while ($num > 0);
```

for Loop

```
for ($distance=50; $distance<=250; $distance+=50)
{
  echo "<tr>
  <td align=\"right\">".$distance."</td>
  <td align=\"right\">".($distance/10)."</td>
  </tr><br>";
}
```

Creating and initializing an array

- Array of discrete values

```
$products = array('Tires',
'Oil', 'Spark plugs');
```

- Array of numbers from 1 to 10

```
$numbers = range(1, 10);
```

- Array of odd numbers from 1 to 10

```
$oddnumbers = range(1, 10, 2);
```

Using Arrays

- Setting a value

```
$products[0] = 'Fuses';
```

- Displaying values

```
echo "$products[0] $products[1]";
```

Loops and Arrays

- for Loop

```
for ($i=0; $i<3; $i++) {
  echo $products[$i]." ";
}
```

- foreach Loop

```
foreach ($products as $current) {
  echo $current." ";
}
```

Keys and Values

- Create an array of key/value pairs

```
$prices=array('Tires'=>100, 'Oil'=>10, 'Plugs'=>4);
```

- Add to array

```
 $prices['WindshieldWiper'] = 8;
```

- Accessing key/value pairs with foreach loop

```
 foreach ($prices as $key => $value) {
     echo $key." – ".$value;
 }
```

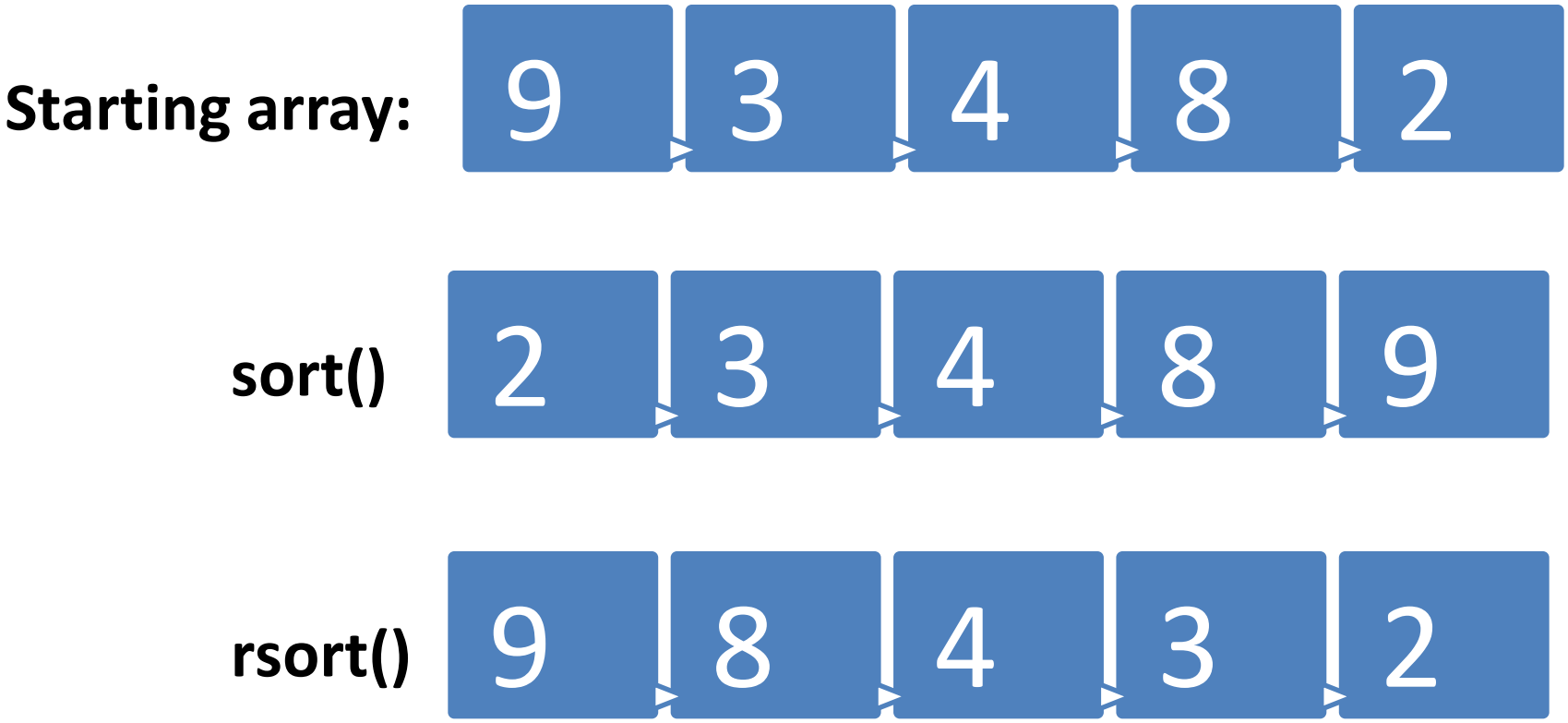
- Accessing key/value pairs with each

```
while ($element = each($prices)) {
    echo $element['key'];
    echo " – ";
    echo $element['value'];
    echo "<br/>";
}
```

Array Operators

Operator	Name	Use	Result
`+`	Union	`$a + $b`	Returns an array containing everything in `$a` and `$b`
`==`	Equality	`$a == $b`	Returns `true if $a` and `$b` have the same key and pairs
`===`	Identity	`$a === $b`	Returns `true if $a` and `$b` have the same key and value pairs the same order
`!=`	Inequality	`$a != $b`	Returns `true if $a` and `$b` are not equal
`<>`	Inequality	`$a <> $b`	Returns `true if $a` and `$b` are not equal
`!==`	Non-identity	`$a !== $b`	Returns `true if $a` and `$b` are not identical

Sorting Arrays



Sorting Key/Value Pairs by Value

Sorting Key/Value Pairs by Key

Other Array Functions

Function	Description
shuffle()	Randomize values
array_reverse()	Create a new array in reverse order
array_push()	Add an item to the end of the array
array_pop()	Remove an item from the end of the array
array_walk()	Execute a function on each value in an array
each(), current, reset(), end(), next(), pos(), prev()	Navigate through an array
count(), sizeof(), array_count_values()	Count elements in an array
extract()	Extract values from key/value array