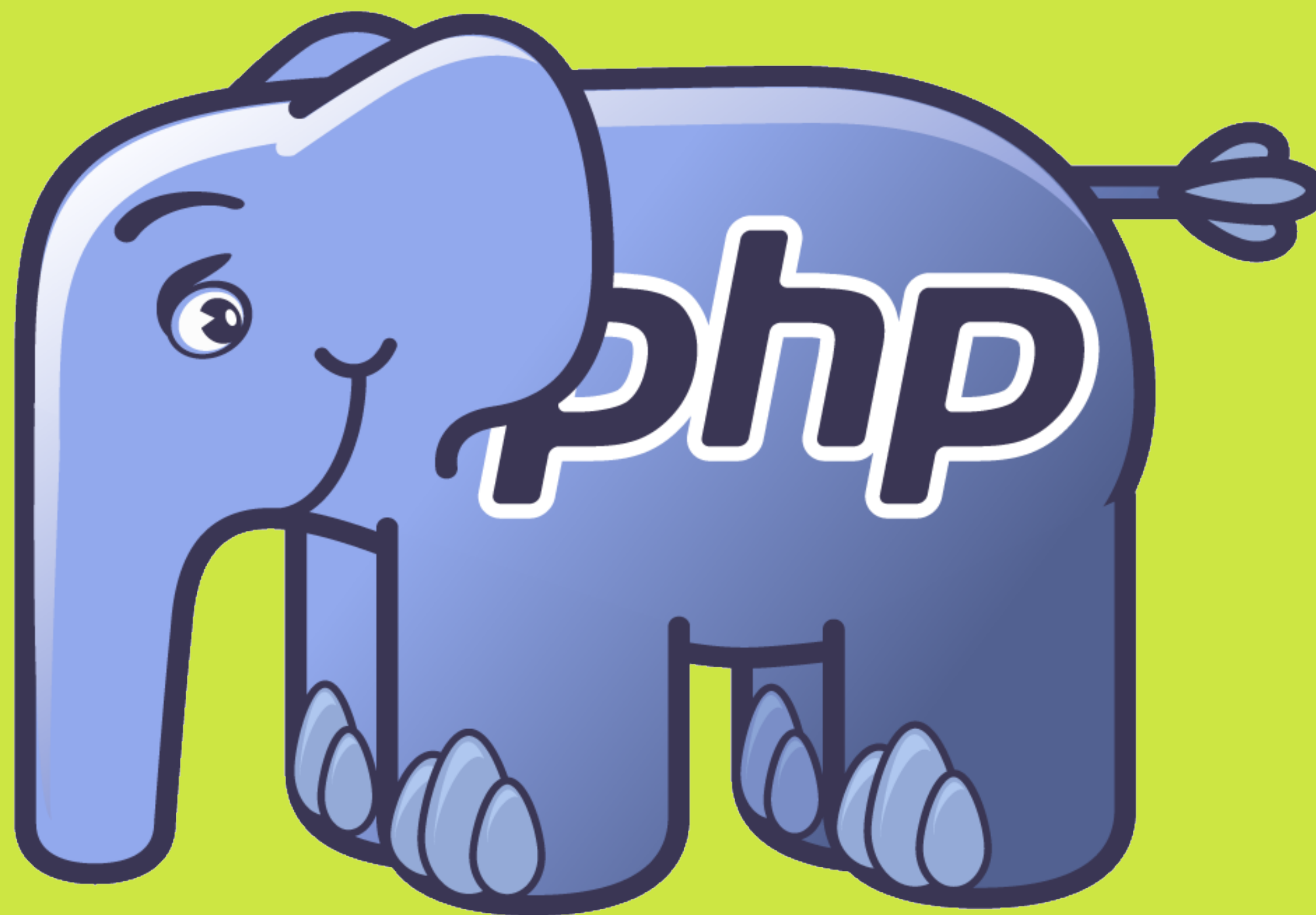




# Fundamentals

VERSION .03





# TERMINOLOGY

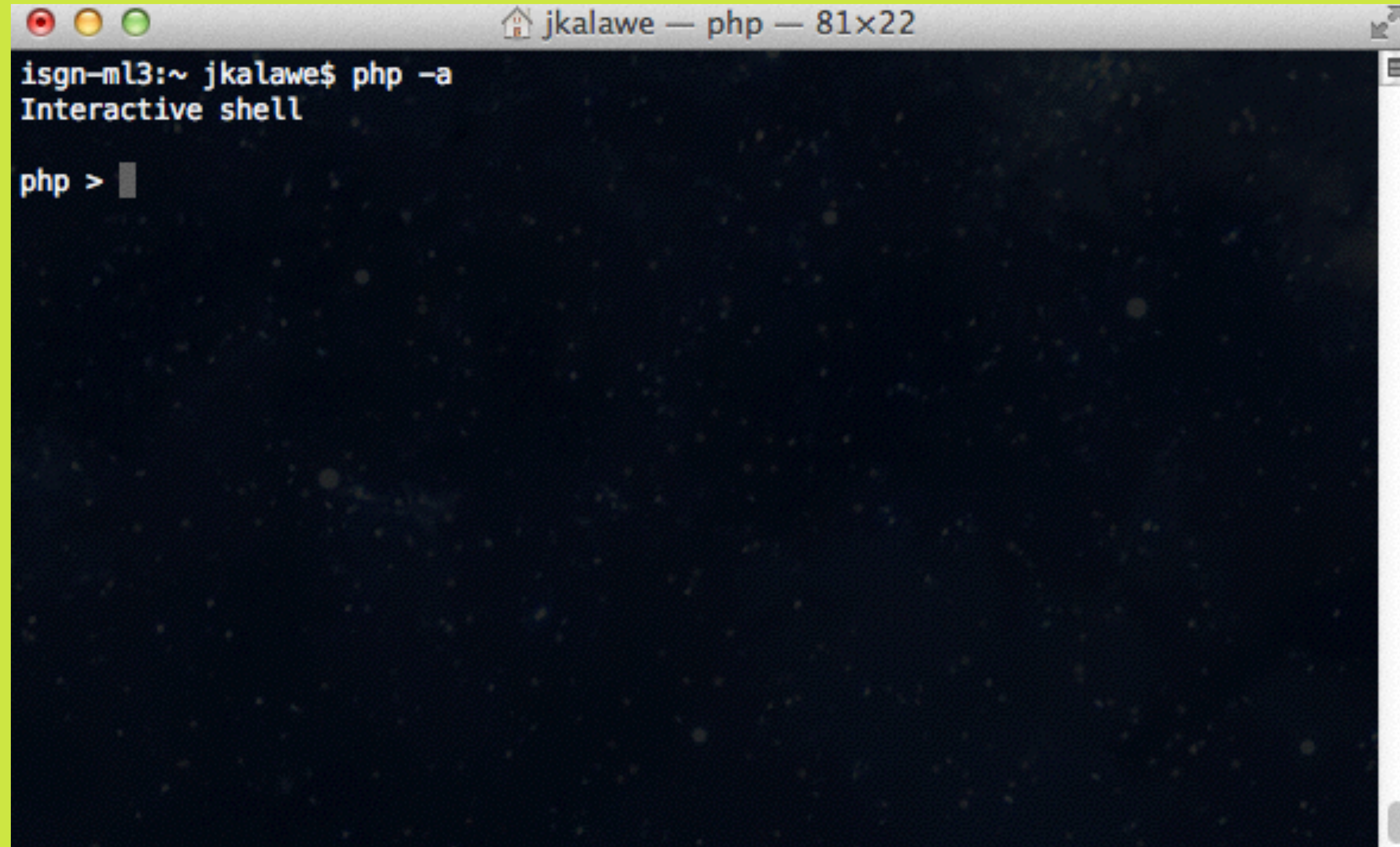
## PHP INTERACTIVE SHELL

Type and execute code in the shell

OPEN YOUR SHELL WINDOW AND TYPE THE FOLLOWING

```
php -a
```



A screenshot of a macOS terminal window. The title bar at the top shows three colored window control buttons (red, yellow, green) on the left, a home icon followed by the text 'jkalawe — php — 81x22' in the center, and a close button on the right. The terminal area has a black background with white text. The first line shows the prompt 'isgn-ml3:~ jkalawe\$' followed by the command 'php -a'. The second line shows the output 'Interactive shell'. The third line shows the prompt 'php >' followed by a cursor. The right side of the terminal window has a vertical scrollbar.

```
isgn-ml3:~ jkalawe$ php -a
Interactive shell

php > 
```

# TERMINOLOGY

EVERY line must end with a semicolon

FOLLOW ALONG

Let's output a text string

```
echo "hello world";
```



# PRIMITIVE DATA TYPES



# TERMINOLOGY

## PRIMITIVE DATA TYPE

Building blocks for any language

OPEN YOUR SHELL WINDOW AND TYPE THE FOLLOWING

```
php -a
```



# TERMINOLOGY

String

Series of characters.

Boolean

True or false.

# TERMINOLOGY

Integer

Whole number: 1,2,3,4

Float

Real number: 1.1, 2.3, -1.1



# VARIABLES



# TERMINOLOGY

VARIABLE

Storage & reference of a value.

MUST START WITH A DOLLAR SIGN

\$test



# FOLLOW ALONG

Assign an integer to a variable

```
$int = 1;
```

Output the value of the variable

```
echo $int;
```

# FOLLOW ALONG

Assign a string to a variable

```
$string = "I don't like this cheese";
```

Output the value of the variable

```
echo $string;
```



# FOLLOW ALONG

Assign a boolean to a variable

```
$bool = TRUE;
```

Output the value of the variable

```
echo $bool;
```

# OPERATORS





# TERMINOLOGY

## OPERATOR

Something that takes a value and outputs another value



# FOLLOW ALONG

## string concatenation

```
$concat = $string . $int . $bool;
```

```
echo $concat;
```

## string concatenation assignment

```
$string = $string .= "next line";
```

```
echo $string;
```



# FOLLOW ALONG

## ARithmetic

```
echo 2+2;
```

```
echo 1*2;
```

## LOGic

```
echo 100 > 50;
```

```
echo 100 >= 50;
```



# conditionals



# TERMINOLOGY

if

Evaluates if expression is true



# FOLLOW ALONG

Let's see if 100 is greater than 50

```
if( 100 > 50) {
```

```
  echo '100 is greater than 50';
```

```
}
```

# TERMINOLOGY

ELSE

Executes when if statement fails.

# FOLLOW ALONG

Let's return a message if the expression fails

```
if( 100 < 50) {
```

```
    echo '100 is greater than 50';
```

```
} else {
```

```
    echo '100 is less than 50';
```

```
}
```