Kill Your Tech Interview

Q1: What is the purpose of `php.ini` file? ☆

Topics: PHP

Answer:

The PHP configuration file, *php.ini*, is the final and most immediate way to affect PHP's functionality. The php.ini file is read each time PHP is initialized.in other words, whenever httpd is restarted for the module version or with each script execution for the CGI version.

Q2: What is the return type of a function that doesn't return anything? ☆

Topics: PHP

Answer:

`void` which mean nothing.

Q3: What is the use of `ini_set()`? ☆

Topics: PHP

Answer:

PHP allows the user to modify some of its settings mentioned in php.ini using ini_set(). This function requires two string arguments. First one is the name of the setting to be modified and the second one is the new value to be assigned to it.

Given line of code will enable the display_error setting for the script if it’s disabled.

`ini_set('display_errors', '1');`

We need to put the above statement, at the top of the script so that, the setting remains enabled till the end. Also, the values set via ini_set() are applicable, only to the current script. Thereafter, PHP will start using the original values from php.ini.

Q4: Implement enqueue and dequeue using only two stacks ☆☆

Topics: JavaScript

Answer:

Enqueue means to add an element, *dequeue* to remove an element.

```
var inputStack = []; // First stack
var outputStack = []; // Second stack
```

```
// For enqueue, just push the item into the first stack
function enqueue(stackInput, item) {
  return stackInput.push(item);
}

function dequeue(stackInput, stackOutput) {
  // Reverse the stack such that the first element of the output stack is the
  // last element of the input stack. After that, pop the top of the output to
  // get the first element that was ever pushed into the input stack
  if (stackOutput.length <= 0) {
    while(stackInput.length > 0) {
      var elementToOutput = stackInput.pop();
      stackOutput.push(elementToOutput);
    }
  }

  return stackOutput.pop();
}
```

Q5: How to check if an object is an array or not? Provide some code. ☆☆

Topics: JavaScript

Answer:

> The best way to find whether an object is instance of a particular class or not using `toString` method from `Object.prototype`

```
var arrayList = [1 , 2, 3];
```

One of the best use cases of type checking of an object is when we do method overloading in JavaScript. For understanding this let say we have a method called `greet` which take one single string and also a list of string, so making our `greet` method workable in both situation we need to know what kind of parameter is being passed, is it single value or list of value?

```
function greet(param) {
  if() {
    // here have to check whether param is array or not
  }
  else {
  }
}
```

However, in above implementation it might not necessary to check type for array, we can check for single value string and put array logic code in else block, let see below code for the same.

```
 function greet(param) {
   if(typeof param === 'string') {
   }
   else {
     // If param is of type array then this block of code would execute
   }
 }
```

Now it's fine we can go with above two implementations, but when we have a situation like a parameter can be `single value`, `array`, and `object` type then we will be in trouble.

Coming back to checking type of object, As we mentioned that we can use `Object.prototype.toString`

```
if(Object.prototype.toString.call(arrayList) === '[object Array]') {
  console.log('Array!');
}
```

If you are using `jQuery` then you can also used jQuery `isArray` method:

```
if($.isArray(arrayList)) {
  console.log('Array');
} else {
  console.log('Not an array');
}
```

FYI jQuery uses `Object.prototype.toString.call` internally to check whether an object is an array or not.

In modern browser, you can also use:

```
Array.isArray(arrayList);
```

`Array.isArray` is supported by Chrome 5, Firefox 4.0, IE 9, Opera 10.5 and Safari 5

Q6: Mention the command to *insert a document* in a database called `school` and collection called `persons`. ☆☆

Topics: MongoDB

Answer:

```
use school;
db.persons.insert( { name: "kadhir", dept: "CSE" } )
```

Q7: Stock maximum profit ☆☆

Topics: JavaScript

Problem:

You will be given a list of stock prices for a given day and your goal is to return the maximum profit that could have been made by buying a stock at the given price and then selling the stock later on.

For example if the input is:

```
[45, 24, 35, 31, 40, 38, 11]
```

then your program should return 16 because if you bought the stock at $24 and sold it at 40$, a profit of $16 was made and this is the largest profit that could be made. If no profit could have been made, return -1.

Solution:

We'll solve the challenge the following way:

1. Iterate through each number in the list.
2. At the ith index, get the `i+1` index price and check if it is larger than the ith index price.
3. If so, set `buy_price = i` and `sell_price = i+1`. Then calculate the profit: `sell_price - buy_price`.
4. If a stock price is found that is cheaper than the current `buy_price`, set this to be the new buying price and continue from step 2.
5. Otherwise, continue changing only the `sell_price` and keep `buy_price` set.

This algorithm runs in linear time, making only one pass through the array, so the running time in the worst case is `O(n)`.

```
function StockPicker(arr) {

  var max_profit = -1;
  var buy_price = 0;
  var sell_price = 0;

  // this allows our loop to keep iterating the buying
  // price until a cheap stock price is found
  var change_buy_index = true;

  // loop through list of stock prices once
  for (var i = 0; i < arr.length-1; i++) {

    // selling price is the next element in list
    sell_price = arr[i+1];

    // if we have not found a suitable cheap buying price yet
    // we set the buying price equal to the current element
    if (change_buy_index) { buy_price = arr[i]; }

    // if the selling price is less than the buying price
    // we know we cannot make a profit so we continue to the
    // next element in the list which will be the new buying price
    if (sell_price < buy_price) {
      change_buy_index = true;
      continue;
    }

    // if the selling price is greater than the buying price
    // we check to see if these two indices give us a better
    // profit then what we currently have
    else {
      var temp_profit = sell_price - buy_price;
      if (temp_profit > max_profit) { max_profit = temp_profit; }
      change_buy_index = false;
    }

  }

  return max_profit;

}

StockPicker([44, 30, 24, 32, 35, 30, 40, 38, 15]);
```

Q8: Determine overlapping numbers in ranges ☆☆

Topics: JavaScript

Problem:

You will be given an array with `5` numbers. The first 2 numbers represent a range, and the next two numbers represent another range. The final number in the array is `X`. The goal of your program is to determine if both ranges overlap by at least `X` numbers. For example, in the array `[4, 10, 2, 6, 3]` the ranges `4` to `10` and `2` to `6` overlap by at least `3` numbers `(4, 5, 6)`, so your program should return `true`. Solve with and without looping.

If the array is `[10, 20, 4, 14, 4]` then the ranges are:

```
10 11 12 13 14 15 16 17 18 19 20
4 5 6 7 8 9 10 11 12 13 14
```

These ranges overlap by at least `4` numbers, namely: `10, 11, 12, 13, 14` so your program should return `true`.

Solution:

With loop:

```
function OverlappingRanges(arr) {

  // keep a count of how many numbers overlap
  var counter = 0;

  // loop through one of the ranges
  for (var i = arr[0]; i < arr[1]; i++) {

    // check if a number within the first range exists
    // in the second range
    if (i >= arr[2] && i <= arr[3]) {
      counter += 1;
    }

  }

  // check if the numbers that overlap is equal to or greater
  // than the last number in the array
  return (counter >= arr[4]) ? true : false;
}

OverlappingRanges([4, 10, 2, 6, 3]);
```

Without loop:

```
function overlapping(input){
  var nums1 = listOfNums(input[0], input[1]);
  var nums2 = listOfNums(input[2], input[3]);
  var overlappingNum = 0;

  if(nums1[0] >= nums2[0] && nums1[0] <= nums2[1]){
    overlappingNum =  nums2[1] - nums1[0] + 1;
  } else {
    overlappingNum =  nums1[1] - nums2[0] + 1;
  }
  if(overlappingNum >= input[4]){
    return true;
  }
}

function listOfNums(a, b){
  var start = a;
  var end = b;
  if(a > b){
    start = b;
    end = a;
  }

  return [a, b];
}

var a = [4, 10, 2, 6, 3];
overlapping(a)
```

Q9: How would you check if a number is an integer? ☆☆

Topics: JavaScript

Answer:

A very simply way to check if a number is a decimal or integer is to see if there is a remainder left when you divide by 1.

```
function isInt(num) {
  return num % 1 === 0;
}

console.log(isInt(4)); // true
console.log(isInt(12.2)); // false
console.log(isInt(0.3)); // false
```

Q10: Can we have multiple `document.ready()` function on the same page? ☆☆

Topics: jQuery

Answer:

YES. We can have any number of document.ready() function on the same page.

Q11: What is the use of jQuery `.each()` function? ☆☆

Topics: jQuery

Answer:

The `$.each()` function is used to iterate over a jQuery object. The `$.each()` function can be used to iterate over any collection, whether it is an object or an array.

Q12: What is `event.PreventDefault()` ? ☆☆

Topics: jQuery

Answer:

The `event.preventDefault()` method stops the default action of an element from happening. For example, Prevents a link from following the URL.

Q13: When should I use `require` vs `include`? ☆☆

Topics: PHP

Answer:

The `require()` function is identical to `include()`, except that it handles errors differently. If an error occurs, the `include()` function generates a warning, but the script will continue execution. The `require()` generates a fatal error, and the script will stop.

My suggestion is to just use `require_once` 99.9% of the time.

Using `require` or `include` instead implies that your code is not **reusable** elsewhere, i.e. that the scripts you're pulling in actually execute code instead of making available a class or some function libraries.

Q14: What is `stdClass` in PHP? ☆☆

Topics: PHP

Answer:

`stdClass` is just a generic 'empty' class that's used when casting other types to objects. `stdClass` is **not** the base class for objects in PHP. This can be demonstrated fairly easily:

```
class Foo{}
$foo = new Foo();
echo ($foo instanceof stdClass)?'Y':'N'; // outputs 'N'
```

It is useful for anonymous objects, dynamic properties, etc.

An easy way to consider the `StdClass` is as an alternative to associative array. See this example below that shows how `json_decode()` allows to get an StdClass instance or an associative array. Also but not shown in this example, `SoapClient::__soapCall` returns an `StdClass` instance.

```
//Example with StdClass
$json = '{ "foo": "bar", "number": 42 }';
$stdInstance = json_decode($json);

echo $stdInstance - > foo.PHP_EOL; //"bar"
echo $stdInstance - > number.PHP_EOL; //42

//Example with associative array
$array = json_decode($json, true);

echo $array['foo'].PHP_EOL; //"bar"
echo $array['number'].PHP_EOL; //42
```

Q15: What are the differences between `die()` and `exit()` functions in PHP? ☆☆

Topics: PHP

Answer:

There's no difference - they are the same. The only advantage of choosing `die()` over `exit()`, might be the time you spare on typing an extra letter.

Q16: What's the difference between `isset()` and `array_key_exists()`? ☆☆

Topics: PHP

Answer:

- `array_key_exists` will tell you if a key exists in an array and complains when `$a` does not exist.
- `isset` will only return `true` if the key/variable exists **and is not `null`**. `isset` doesn't complain when `$a` does not exist.

Consider:

```
$a = array('key1' => 'Foo Bar', 'key2' => null);

isset($a['key1']);             // true
array_key_exists('key1', $a);  // true

isset($a['key2']);             // false
array_key_exists('key2', $a);  // true
```

Q17: What is the differences between `$a != $b` and `$a !== $b`? ☆☆

Topics: PHP

Answer:

`!=` means *inequality* (TRUE if $aisnotequalto$b) and `!==` means *non-identity* (TRUE if $aisnotidenticalto$b).

Q18: What is *PDO* in PHP? ☆☆

Topics: PHP

Answer:

PDO stands for PHP Data Object.

It is a set of PHP extensions that provide a core PDO class and database, specific drivers. It provides a vendor-neutral, lightweight, data-access abstraction layer. Thus, no matter what database we use, the function to issue queries and fetch data will be same. It focuses on data access abstraction rather than database abstraction.

Q19: How do you check if the react native app is in debug or release build? ☆☆

Topics: React Native

Answer:

You can use this command

```
if (__DEV__) {
    console.log('I am in debug');
}
```

Q20: What will be the output of following snippet? ☆☆

Topics: React Native

Problem:

```
const ComponentScreen = () => {
const someArray = ['1', '2', '3']
  return (
    <View>
      <Text>{someArray}</Text>
    </View>
  )
 }
 export default ComponentScreen;
```

Solution:

`123`