Exemplo básico de cURL

Uma vez que você tenha compilado o PHP com suporte a cURL, você pode começar a usar as funções cURL. A ideia por trás das funções cURL é que você inicie uma sessão cURL usando curl_init(), permitindo que você configure suas opções para a transferência através da curl_setopt(), podendo assim executar a sessão com curl_exec() e finalizá-la usando curl_close(). Aqui está um exemplo que usa as funções cURL para pegar o conteúdo da página exemplo.com.br e colocar em um arquivo:

Exemplo #1 Usando o módulo cURL do PHP para pegar o conteúdo de exemplo.com.br

```php
<?php

$ch = curl_init("http://www.exemplo.com.br/");
$fp = fopen("pagina_exemplo.txt", "w");

curl_setopt($ch, CURLOPT_FILE, $fp);
curl_setopt($ch, CURLOPT_HEADER, 0);

curl_exec($ch);
if(curl_error($ch)) {
    fwrite($fp, curl_error($ch));
}
curl_close($ch);
fclose($fp);
?>
```

https://www.php.net/manual/pt_BR/curl.examples-basic.php

Utilizando cURL com PHP

09/07/2018 Categorias: Ferramentas, PHP, Tags: curl,

Em nosso artigo anterior mostramos como utilizar a biblioteca cURL de maneira geral, rodando em linha de comando no linux. Neste artigo iremos mostrar como usar o cURL em conjunto com o **PHP**.

Diversas vezes precisamos **realizar requisições** ou transferir data em nosso **código backend**. Uma das maneiras mais completas de se fazer isso é utilizando o cURL. Além de requisições simples como **GET** e **POST** com ele podemos utilizar vários protocolos como IMAP, POP, FTP e outros.

Básico

Para começar nós devemos chamar a função `curl_init()` que retorna um recurso cURL. Esta função aceita um único parâmetro que é a URL para qual a requisição vai ser enviada. Esta URL pode ser informada depois em uma configuração.

```
$curl = curl_init();
```

Configurações

Assim que iniciarmos o nosso recurso cURL podemos começar a passar parâmetros para ele, assim configurando nossa requisição. Abaixo listamos alguns parâmetros básicos:

- `CURLOPT_RETURNTRANSFER` - Retorna a resposta como uma string ao invés de printar na tela
- `CURLOPT_CONNECTTIMEOUT` - Segundos tentando conectar até o timeout
- `CURLOPT_TIMEOUT` - Segundos limite para execução do cURL
- `CURLOPT_USERAGENT` - String contendo um user-agent para a requisição
- `CURLOPT_URL` - URL para enviar a requisição
- `CURLOPT_PORT` - Para informar uma porta
- `CURLOPT_HTTPHEADER` - Cabeçalhos da requisição
- `CURLOPT_POST` - Envia a requisição como POST
- `CURLOPT_POSTFIELDS` - Array de informações enviadas como POST

Nós podemos informar um parâmetro usando a função `curl_setopt()`, assim como no exemplo abaixo:

```
$curl = curl_init();
curl_setopt($curl, CURLOPT_URL, 'http://exemplo.com.br');
```

É possível informar diversos parâmetros ao mesmo tempo usando a função `curl_setopt_array()` e passando um array de opções:

```
$curl = curl_init();
curl_setopt_array($curl, [
    CURLOPT_RETURNTRANSFER => 1,
    CURLOPT_URL => 'http://exemplo.com.br'
]);
```

Enviando a requisição

Assim que todas as opções estiverem configuradas e sua requisição pronta para ser enviada você deve usar a função `curl_exec()` para executar. Os retornos podem ser os seguintes:

- `false` - Se houve algum erro ao executar a requisição
- `true` - Se a requisição foi realizada e a opção `CURLOPT_RETURNTRANSFER` estiver setada para `false`
- **A resposta** - Se a opção acima estiver setada como `true` o retorno será a resposta da requisição realizada

Utilizando o exemplo anterior nós podemos pegar o resultado da seguinte maneira:

```
$result = curl_exec($curl);
```

Com o resultado sendo o retorno da requisição, que pode ser um JSON, uma string ou mesmo HTML

Fechando a requisição

Após realizar a requisição e utilizar os dados retornados é interessante você fechar o cURL para limpar alguns recursos do sistema (como memória RAM), para isto basta usar a função `curl_close()`

Exemplo - Requisição GET com cURL e PHP

```
// Cria o cURL
$curl = curl_init();
// Seta algumas opções
curl_setopt_array($curl, [
    CURLOPT_RETURNTRANSFER => 1,
    CURLOPT_URL => 'http://exemplo.com.br/?item1=valor&item2=valor'
]);
// Envia a requisição e salva a resposta
$response = curl_exec($curl);
// Fecha a requisição e limpa a memória
curl_close($curl);
```

Exemplo - Requisição POST com cURL e PHP

```
// Cria o cURL
$curl = curl_init();
// Seta algumas opções
curl_setopt_array($curl, [
    CURLOPT_RETURNTRANSFER => 1,
    CURLOPT_URL => 'http://exemplo.com.br',
    CURLOPT_POST => 1,
    CURLOPT_POSTFIELDS => [
        item1 => 'valor',
        item2 => 'valor'
    ]
]);
// Envia a requisição e salva a resposta
```

```
$response = curl_exec($curl);
// Fecha a requisição e limpa a memória
curl_close($curl);
```

Erros

É importante sempre tratar possíveis erros que aconteçam na requisição. Para isso existem duas funções do cURL:

- `curl_error()` - Retorna uma string com uma mensagem de erro, se a string estiver em branco, nenhum erro aconteceu
- `curl_errno()` - Retorna um código de erro

--

Artigo baseado em http://codular.com/curl-with-php

https://codecommit.com.br/curl-com-php

5 PHP cURL examples

Published September 06, 2015

Working on the server side does not necessarily imply that all the required information needs to be present on the database of the site on which we are working. As a matter of fact, growing parts of the information that is used on the server side comes from external sources, often via an API that allows some kind of service to provide information to the website without having access to the database on the remote site. To utilize this information, we can use the **cURL** built-in PHP extension.

cURL is a PHP extension, that allows us to receive and send information via the URL syntax. By doing so, cURL makes it easy to communicate between different websites and domains. This tutorial includes 5 common cases for the use of cURL, and they include:

1. Downloading the content of a website
2. Downloading a file from a website
3. Auto form submission
4. Authentication
5. Use of cookies

Joseph Benharosh is a full stack web developer and the author of the eBook The essentials of object oriented PHP.

# How does the cURL extension work?

cURL works by sending a request to a web site, and this process includes the following four parts:

1. Initialization.

```
$handle = curl_init();
```

2. Setting the options. There are many options, for example, an option that defines the URL.

```
curl_setopt($handle, CURLOPT_URL, $url);
```

3. Execution with curl_exec().

```
$data = curl_exec($handle);
```

4. Releasing the cURL handle.

```
curl_close($handle);
```

The second part is the most interesting because it allows us to define how cURL works in a highly accurate manner, by using the many options it has to offer.

# 1. How to download the contents of a remote website to a local file?

In order to download the contents of a remote web site, we need to define the following options:

CURLOPT_URL- Defines the remote URL.

CURLOPT_RETURNTRANSFER- Enables the assignment of the data that we download from the remote site to a variable. In this example, we assign the data into the variable $output.

```
<?php
$handle = curl_init();

$url = "https://www.ladygaga.com";

// Set the url
curl_setopt($handle, CURLOPT_URL, $url);
// Set the result output to be a string.
curl_setopt($handle, CURLOPT_RETURNTRANSFER, true);

$output = curl_exec($handle);

curl_close($handle);

echo $output;
```

When we print the value of the variable $output to the screen, we will see the local version of the web site for the world's best singer.

The options can be written more compactly using curl_setopt_array(), which is a cURL function that convenes the options into an array.

```
curl_setopt_array($handle,
  array(
      CURLOPT_URL            => $url,
      CURLOPT_RETURNTRANSFER => true
  )
);
```

# 2. How to download a file from a remote site using cURL?

A remote file can be downloaded to our server, if the option CURLOPT_ FILE is set. For example, the following code downloads the book "The Divine Comedy" from Project Gutenberg into a the_divine_comedy.html file on our server:

```
<?php
// The distant site url.
$url = "https://www.gutenberg.org/files/46852/46852-h/46852-h.htm";
// The file on our server.
$file = __DIR__ . DIRECTORY_SEPARATOR . "the_divine_comedy.html";
$handle = curl_init();

// Open the file on our server for writing.
$fileHandle = fopen($file, "w");
```

```
curl_setopt_array($handle,
  array(
     CURLOPT_URL           => $url,
      CURLOPT_FILE => $fileHandle,
  )
);

$data = curl_exec($handle);

curl_close($handle);

fclose($fileHandle);
```

# Handling the returned response

In order to get the parameters of the response for it to be monitored and debugged, we need to set the option CURLOPT_HEADER. For example:

```
<?php
$url = ""https://www.gutenberg.org/files/41537/41537-h/41537-h.htm";

$file = __DIR__ . DIRECTORY_SEPARATOR . "the_divine_comedy.html";

$handle = curl_init();

$fileHandle = fopen($file, "w");

curl_setopt_array($handle,
  array(
    CURLOPT_URL => $url,
    CURLOPT_FILE  => $fileHandle,
    CURLOPT_HEADER => true
  )
);

$data = curl_exec($handle);
```

To get additional information about the request, we use the curl_getinfo command that enables us to receive important technical information about the response, including the status code (200 for success) and the size of the downloaded file.

```
$responseCode   =
    curl_getinfo($handle,
        CURLINFO_HTTP_CODE
);

$downloadLength =
    curl_getinfo($handle,
         CURLINFO_CONTENT_LENGTH_DOWNLOAD
);
```

In addition, we can also use the commands: curl_error and curl_errno to debug the response and receive informative error messages.

```
if(curl_errno($handle))
{
  print curl_error($handle);
}
```

Let's see the full code:

```php
<?php
$url = "https://www.gutenberg.org/files/46852/46852-h/46852-h.htm";

$file = __DIR__ . DIRECTORY_SEPARATOR . "the_divine_comedy.html";

$handle = curl_init();

$fileHandle = fopen($file, "w");

curl_setopt_array($handle,
  array(
    CURLOPT_URL    => $url,
    CURLOPT_FILE   => $fileHandle,
    CURLOPT_HEADER => true
  )
);

$data = curl_exec($handle);

$responseCode   = curl_getinfo($handle, CURLINFO_HTTP_CODE);

$downloadLength = curl_getinfo($handle, CURLINFO_CONTENT_LENGTH_DOWNLOAD);

if(curl_errno($handle))
{
  print curl_error($handle);
}
else
{
  if($responseCode == "200") echo "successful request";

  echo " # download length : " . $downloadLength;

  curl_close($handle);

  fclose($fileHandle);
}
```

# 3. How to submit forms with cURL?

Until this moment, we have demonstrated the use of the GET method of HTTP (which is generally used to watch and download content). cURL can also make use of the POST method of HTTP in order to submit forms.

In order to demonstrate form submission with cURL, we need to create the following two files:

1. index.php in which we put the cURL script.
2. form.php in which we put the form to be submitted.

The form.php will be in reality, found on a remote server (although, for the sake of the example, both files may be placed on the same server). Also, for the example, we will use a form with 3 fields: firstName, lastName and submit.

```php
<?php
if(isset($_POST["submit"]))
{
  echo "Full name is " . $_POST["firstName"] .
     "  " . $_POST["lastName"];
  exit;
}
?>

<html>
<body>

<form method = "POST" action = "" >
  <input  name="firstName"  type="text">
  <input  name="lastName"  type="text">
  <input  type="submit"  name="submit"  value="שלח" >
</form>
</body>
</html>
```

To submit the form, the following options need to be set:

1. CURLOPT_POST– Sets the request to be in a post mode.
2. CURLOPT_POSTFIELDS- Receives the associative array of the fields that we want to post. The array keys are named after the name of the form fields.

```
<?php
$handle = curl_init();

$url = "https://localhost/curl/theForm.php";

// Array with the fields names and values.
// The field names should match the field names in the form.

$postData = array(
  'firstName' => 'Lady',
  'lastName'  => 'Gaga',
  'submit'    => 'ok'
);

curl_setopt_array($handle,
  array(
     CURLOPT_URL => $url,
     // Enable the post response.
    CURLOPT_POST       => true,
    // The data to transfer with the response.
    CURLOPT_POSTFIELDS => $postData,
    CURLOPT_RETURNTRANSFER     => true,
  )
);

$data = curl_exec($handle);

curl_close($handle);

echo $data;
```

# 4. How to perform basic HTTP authentication with cURL?

In order to authenticate with cURL, the following 3 options need to be set:

1. CURLOPT_HTTPAUTH
2. CURLOPT_USERPWD– Through which we define the username and password.
3. CURLOPT_RETURNTRANSFER

Let's see the code:

```
curl_setopt_array($handle,
  array(
    CURLOPT_URL => $url,
   CURLOPT_HTTPAUTH => CURLAUTH_ANY,
   CURLOPT_USERPWD  => "$username:$password",
   CURLOPT_RETURNTRANSFER   => true,
  )
);
```

# 5. How to handle cookies with cURL?

The use of cookies allows a website to identify returning visitors and authenticated users. To this end, cURL provides us with a mechanism through which we can save cookies.

The two main options that allow us to handle cookies are:

1. CURLOPT_COOKIEJAR– Defines the file required to write the cookies.
2. CURLOPT_COOKIEFILE– Defines the file from which the cookies are to be read.

The following code example writes the cookies into a cookie.txt file on the first visit, and then reads the data in later visits.

```
<?php

$handle = curl_init();

$url = "https://www.ladygaga.com/artrave-the-artpop-ball";

$file = __DIR__ . DIRECTORY_SEPARATOR . "cookie.txt";

curl_setopt_array($handle,
  array(
    CURLOPT_URL => $url,
     // The file to which the cookies need to be written.
    CURLOPT_COOKIEFILE => $file,
    // The file freom which the cookies need to be read.
    CURLOPT_COOKIEJAR  => $file,
    CURLOPT_RETURNTRANSFER     => true,
  )
);

$data = curl_exec($handle);

curl_close($handle);
```

Conclusion

Using PHP's cURL extension provides us with a convenient way to communicate with other web sites, particularly with APIs that are provided by a third party. In the next tutorial, we will learn how to request for private details in the name of users that sign in to our website with their GitHub account. It will be done by using Github's API, and with the help of cURL. The tutorial will be a good starting point for learning how to make a social login with any social network.

https://phpenthusiast.com/blog/five-php-curl-examples

PHP Curl Tutorial: Everything You Need To Know

Published on Jul 29,2019 2.5K Views

edureka

-

CURL is a way from where we can hit a URL from our code to get an HTML response from it. It allows you to connect with other URLs and use their responses in our code. In this PHP CURL Tutorial we will be exploring this concept in detail.

Following pointers will be covered in this article and in detail,

- What is CURL?
- Download contents of a remote website to a local file

So let get started with PHP CURL tutorial

PHP CURL tutorial

What is CURL?

CURL stands for Client URL and it is a library that lets you make HTTP requests in PHP

First of all, we create a curl resource with curl_init() function

```
$curl=curl_init();
```

Second step is to set curl options using curl_setopt() function

```
curl_setopt($curl, CURLOPT_URL, 'https://www.edureka.co');
```

Third step is to run curl or execute HTTP requests with curl_exec()

```
curl_exec($curl);
```

Fourth step is to close curl resource with curl_close() to free up the resources

```
curl_close($curl);
```

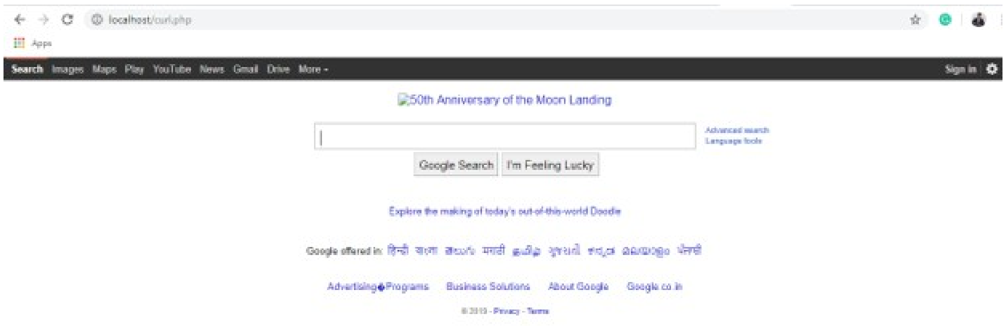
That’s how we create curl requests

```
<?php
$curl=curl_init();
/*curl is going to be datatype curl resource. when we have a curl
resource, we are able to use functions that are specifically
designed for that resource like curl_setopt*/
curl_setopt($curl, CURLOPT_URL, 'https://www.google.com');
```

```
curl_exec($curl);
?>
```

Output:

After loading this file in the browser, google is loaded using curl request


localhost/curl.php
Apps
Search Images Maps Play YouTube News Gmail Drive More
Sign in
50th Anniversary of the Moon Landing
Advanced search
Language tools
Google Search
I'm Feeling Lucky
Explore the making of today's out-of-this-world Doodle
Google offered in: हिन्दी বাংলা తెలుగు मराठी தமிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ
Advertising Programs
Business Solutions
About Google
Google.co.in
© 2019 - Privacy - Terms

For the https://www.edureka.co/, they actually use the HTTPS protocol and in order to load this, we are going to set our CURLOPT_SSL_VERIFYPEER to false which is one of the options available in curl.

https://www.edureka.co/blog/php-curl-tutorial/

PHP | cURL

- Difficulty Level : Hard
- Last Updated : 04 Oct, 2021

The cURL stands for 'Client for URLs', originally with URL spelled in uppercase to make it obvious that it deals with URLs. It is pronounced as 'see URL'. The cURL project has two products libcurl and curl.

- **libcurl:** A free and easy-to-use client-side URL transfer library, supporting FTP, TPS, HTTP, HTTPS, GOPHER, TELNET, DICT, FILE, and LDAP. libcurl supports TTPS certificates, HTTP POST, HTTP PUT, FTP uploading, kerberos, HTTP based upload, proxies, cookies, user & password authentication, file transfer resume, HTTP proxy tunneling and many more. libcurl is free, thread-safe, IPv6 compatible, feature rich, well supported and fast.
- **curl:** A command line tool for getting or sending files using URL syntax. Since curl uses libcurl, it supports a range of common internal protocols, currently including HTTP, HTTPS, FTP, FTPS, GOPHER, TELNET, DICT, and FILE.

What is PHP/cURL?
The module for PHP that makes it possible for PHP programs to access curl functions within PHP. cURL support is enabled in PHP, the phpinfo() function will display in its output. You are requested to check it before writing your first simple program in PHP.

```
<?php

phpinfo();

?>
```

Simple Uses: The simplest and most common request/operation made using HTTP is to get a URL. The URL itself can refer to a webpage, an image or a file. The client issues a GET request to the server and receives the document it asked for.
Some basic cURL functions:

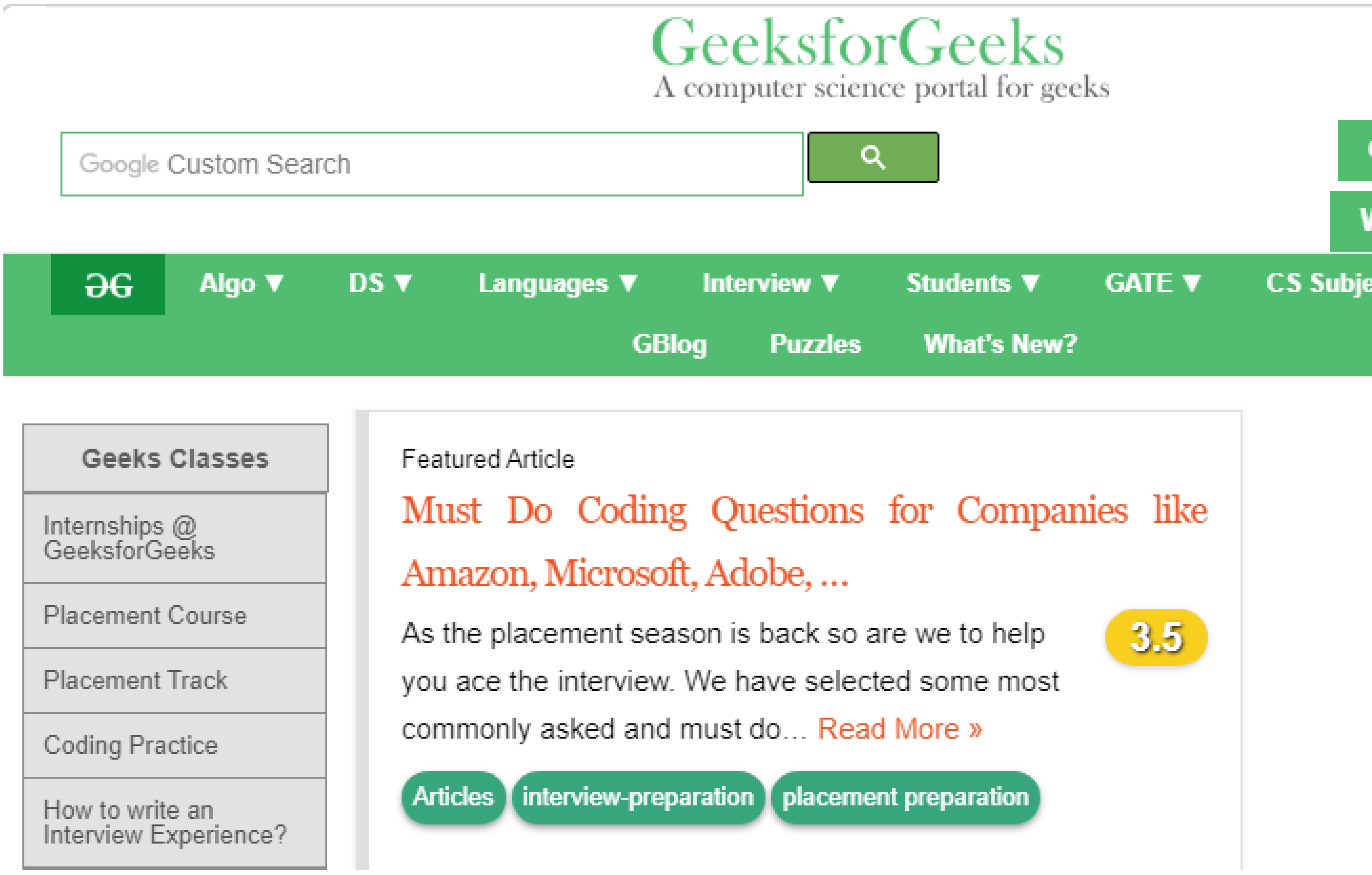
- The *curl_init()* function will initialize a new session and return a cURL handle.
- *curl_exec($ch)* function should be called after initialize a cURL session and all the options for the session are set. Its purpose is simply to execute the predefined CURL session (given by ch).
- *curl_setopt($ch, option, value)* set an option for a cURL session identified by the ch parameter. Option specifies which option is to set, and value specifies the value for the given option.
- *curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1)* return page contents. If set 0 then no output will be returned.

- *curl_setopt($ch, CURLOPT_URL, $url)* pass URL as a parameter. This is your target server website address. This is the URL you want to get from the internet.
- *curl_exec($ch)* grab URL and pass it to the variable for showing output.
- *curl_close($ch)* close curl resource, and free up system resources.

Example:

```
<?php

// From URL to get webpage contents.
$url = "https://www.geeksforgeeks.org/";

// Initialize a CURL session.
$ch = curl_init();

// Return Page contents.
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

//grab URL and pass it to the variable.
curl_setopt($ch, CURLOPT_URL, $url);

$result = curl_exec($ch);

echo $result;

?>
```

Output:


GeeksforGeeks
A computer science portal for geeks

Google Custom Search

Algo ▼ DS ▼ Languages ▼ Interview ▼ Students ▼ GATE ▼ CS Subje
GBlog Puzzles What's New?

Geeks Classes
Internships @ GeeksforGeeks
Placement Course
Placement Track
Coding Practice
How to write an Interview Experience?

Featured Article
Must Do Coding Questions for Companies like Amazon, Microsoft, Adobe, ...
3.5
As the placement season is back so are we to help you ace the interview. We have selected some most commonly asked and must do… Read More »
Articles interview-preparation placement preparation

Reference: http://php.net/manual/en/book.curl.php

https://www.geeksforgeeks.org/php-curl/