# HttpClient

网络爬虫就是用程序帮助我们访问网络上的资源，我们一直以来都是使用HTTP协议访问互联网的网页，网络爬虫需要编写程序，在这里使用同样的HTTP协议访问网页。

这里我们使用Java的HTTP协议客户端 HttpClient这个技术，来实现抓取网页数据。

## GET请求

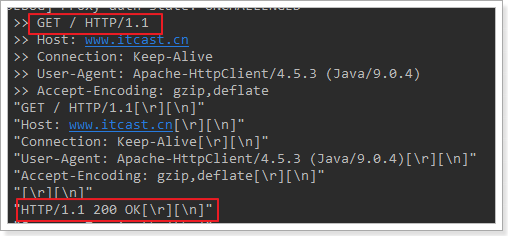
访问传智官网，请求url地址：

<http://www.itcast.cn/>

cn.itcast.crawler.test.HttpGetTest

public static void main(String[] args) throws IOException {  
 //创建HttpClient对象  
 CloseableHttpClient httpClient = HttpClients.*createDefault*();  
  
 //创建HttpGet请求  
 HttpGet httpGet = new HttpGet("http://www.itcast.cn/");  
  
 CloseableHttpResponse response = null;  
 try {  
 //使用HttpClient发起请求  
 response = httpClient.execute(httpGet);  
  
 //判断响应状态码是否为200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 //如果为200表示请求成功，获取返回数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 //打印数据长度  
 System.*out*.println(content);  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 httpClient.close();  
 }  
 }  
}

请求结果



## 带参数的GET请求

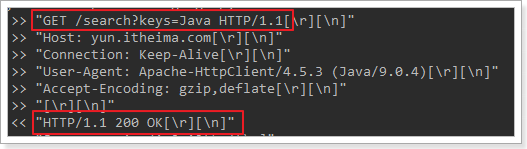
在传智中搜索学习视频，地址为：

<http://yun.itheima.com/search?keys=Java>

cn.itcast.crawler.test.HttpGetParamTest

public static void main(String[] args) throws IOException {  
 //创建HttpClient对象  
 CloseableHttpClient httpClient = HttpClients.*createDefault*();  
  
 //创建HttpGet请求，带参数的地址https://www.baidu.com/s?wd=HttpClient  
 String uri = "http://yun.itheima.com/search?keys=Java";  
 HttpGet httpGet = new HttpGet(uri);  
  
 CloseableHttpResponse response = null;  
 try {  
 //使用HttpClient发起请求  
 response = httpClient.execute(httpGet);  
  
 //判断响应状态码是否为200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 //如果为200表示请求成功，获取返回数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 //打印数据长度  
  
 System.*out*.println(content);  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 httpClient.close();  
 }  
 }  
}

请求结果



## POST请求

使用POST访问传智官网，请求url地址：

<http://www.itcast.cn/>

public static void main(String[] args) throws IOException {  
 //创建HttpClient对象  
 CloseableHttpClient httpClient = HttpClients.*createDefault*();  
  
 //创建HttpGet请求  
 HttpPost httpPost = new HttpPost("http://www.itcast.cn/");  
  
 CloseableHttpResponse response = null;  
 try {  
 //使用HttpClient发起请求  
 response = httpClient.execute(httpPost);  
  
 //判断响应状态码是否为200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 //如果为200表示请求成功，获取返回数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 //打印数据长度  
 System.*out*.println(content);  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 httpClient.close();  
 }  
 }  
}

请求结果：



## 带参数的POST请求

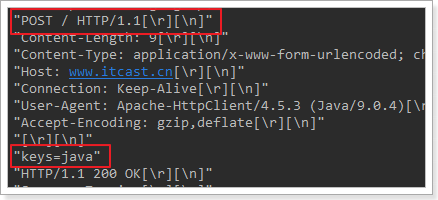
在传智中搜索学习视频，使用POST请求，url地址为：

<http://yun.itheima.com/search>

url地址没有参数，参数keys=java放到表单中进行提交

public static void main(String[] args) throws IOException {  
 //创建HttpClient对象  
 CloseableHttpClient httpClient = HttpClients.*createDefault*();  
  
 //创建HttpGet请求  
 HttpPost httpPost = new HttpPost("http://www.itcast.cn/");  
  
  
 //声明存放参数的List集合  
 List<NameValuePair> params = new ArrayList<NameValuePair>();  
 params.add(new BasicNameValuePair("keys", "java"));  
  
 //创建表单数据Entity  
 UrlEncodedFormEntity formEntity = new UrlEncodedFormEntity(params, "UTF-8");  
  
 //设置表单Entity到httpPost请求对象中  
 httpPost.setEntity(formEntity);  
  
 CloseableHttpResponse response = null;  
 try {  
 //使用HttpClient发起请求  
 response = httpClient.execute(httpPost);  
  
 //判断响应状态码是否为200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 //如果为200表示请求成功，获取返回数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 //打印数据长度  
 System.*out*.println(content);  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 httpClient.close();  
 }  
 }  
}

请求结果



## 连接池

如果每次请求都要创建HttpClient，会有频繁创建和销毁的问题，可以使用连接池来解决这个问题。

测试以下代码，并断点查看每次获取的HttpClient都是不一样的。

public static void main(String[] args) {  
 PoolingHttpClientConnectionManager cm = new PoolingHttpClientConnectionManager();  
  
 // 设置最大连接数  
 cm.setMaxTotal(200);  
  
 // 设置每个主机的并发数  
 cm.setDefaultMaxPerRoute(20);  
  
 *doGet*(cm);  
  
 *doGet*(cm);  
  
}  
  
private static void doGet(PoolingHttpClientConnectionManager cm) {  
 CloseableHttpClient httpClient = HttpClients.*custom*().setConnectionManager(cm).build();  
  
 HttpGet httpGet = new HttpGet("http://www.itcast.cn/");  
  
 CloseableHttpResponse response = null;  
  
 try {  
 response = httpClient.execute(httpGet);  
  
 // 判断状态码是否是200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 // 解析数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 System.*out*.println(content.length());  
 }  
  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 //不能关闭HttpClient  
 //httpClient.close();  
 }  
 }  
}

## 请求参数

有时候因为网络，或者目标服务器的原因，请求需要更长的时间才能完成，我们需要自定义相关时间

public static void main(String[] args) throws IOException {  
 //创建HttpClient对象  
 CloseableHttpClient httpClient = HttpClients.*createDefault*();  
  
 //创建HttpGet请求  
 HttpGet httpGet = new HttpGet("http://www.itcast.cn/");  
  
 //设置请求参数  
 RequestConfig requestConfig = RequestConfig.*custom*()  
 .setConnectTimeout(1000)//设置创建连接的最长时间  
 .setConnectionRequestTimeout(500)//设置获取连接的最长时间  
 .setSocketTimeout(10 \* 1000)//设置数据传输的最长时间  
 .build();  
  
 httpGet.setConfig(requestConfig);  
  
 CloseableHttpResponse response = null;  
 try {  
 //使用HttpClient发起请求  
 response = httpClient.execute(httpGet);  
  
 //判断响应状态码是否为200  
 if (response.getStatusLine().getStatusCode() == 200) {  
 //如果为200表示请求成功，获取返回数据  
 String content = EntityUtils.*toString*(response.getEntity(), "UTF-8");  
 //打印数据长度  
 System.*out*.println(content);  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 //释放连接  
 if (response == null) {  
 try {  
 response.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 httpClient.close();  
 }  
 }  
}