**智慧校园3.0\_CAS集成文档**

建设单位：广东轻工职业技术学院

2020年12月

**文档修订历史**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **修订时间** | **修订说明** | **修订人** |
| V1.0 | 2020-12-11 | 新增 | 莫纯润 |

# 集成总说明

## 版本说明

客户端版本建议使用3.5.1版本，java demo可参教springboot集成

## CAS服务端地址

* 认证地址：https://sso.gdip.edu.cn/authserver
* 退出地址：https://sso.gdip.edu.cn/authserver/logout

## CAS客户端集成说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **客户端** | **根据官方包改动后的客户端版本** | **官方版本** | **环境** | **支持单点登出** | **备注** |
| java | Cas-client-core-3.2.1 | 3.2.1 | Jdk>=1.5 | 是 | 1.根据官方的casclient包进行了改动  2.主要改动地方为增加http的支持 |
| Cas-client-jdk13 |  | Jdk<=1.4 | 是 |
| php | phpcas | 1.3.2 | PHP>=5.0 | 是 | 官方对phpcas包进行了封装，下载此包，直接解压便可以使用，参见最新的php版本集成。 |
| asp | 代码手写 |  | 无限制 | 否 | 1.官网未发现相关的封装包或者其他的方式。  2.通过asp的代码组合进行登陆，参见asp例子。 |
| .net | DotNetCasClient | 1.0.2 | .framework>=2.0 | 是 | 1.对DotNetCasClient进行了更改。  2.主要涉及到用户属性这个这个部分 attributes,（主要是因为官网提供的例子无法支持属性的读取） |
| Perl | 代码手写 |  |  |  | 1.官网未发现相关的封装包或者其他的方式。 |
| Apache | 代码手写 |  |  |  | 1.官网未发现相关的封装包或者其他的方式。 |
| uPortal | 代码手写 |  |  |  | 1.官网未发现相关的封装包或者其他的方式。 |

# 客户端集成说明

## JAVA客户端集成

### Springboot集成

* 参考例子：springboot集成.zip

### 非springboot方式集成

* 参考例子：java.zip

### java认证接口集成

代理认证配置完后，均需至统一身份管理平台授权访问。先登录身份认证管理平台，在认证管理的认证应用模板，添加需代理认证的应用；添加完应用后，需给相应的组或者帐号授权，允许其访问该应用。

#### 拷贝提供的jar

将“java/lib”下的cas-client-core-3.2.1.jar文件拷贝到客户端应用所在服务器中，并将存放这些jar包的路径设置到应用的classpath中。比如WebContent\WEB-INF\lib中

#### 修改web.xml文件

在第三方应用的web.xml中添加以下内容：（加色部分需要修改为实际参数）

|  |
| --- |
| <!-- ==================== 单点登录开始 ================ -->  <!-- 用于单点退出，该监听器和过滤器用于实现单点登出功能，可选配置 -->  <listener>  <listener-class>org.jasig.cas.client.session.SingleSignOutHttpSessionListener</listener-class>  </listener>  <filter>  <filter-name>CAS Single Sign Out Filter</filter-name>  <filter-class>org.jasig.cas.client.session.SingleSignOutFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>CAS Single Sign Out Filter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- 该过滤器负责用户的认证工作，必须启用它 -->  <filter>  <filter-name>CASFilter</filter-name>  <filter-class>org.jasig.cas.client.authentication.AuthenticationFilter</filter-class>  <init-param>  <param-name>casServerLoginUrl</param-name>  <!--这里的server是CAS服务端的登录地址,login为固定值-->  <param-value> https://sso.gdip.edu.cn/authserver/login</param-value>  </init-param>  <init-param>  <param-name>serverName</param-name>  <!--这里是应用地址，注意是域名:端口或者ip:端口-->  <param-value>http://app:8080</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>CASFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- 该过滤器负责对Ticket的校验工作，必须启用它 -->  <filter>  <filter-name>CAS Validation Filter</filter-name>  <filter-class>org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter</filter-class>  <init-param>  <param-name>casServerUrlPrefix</param-name>  <!--这里的server是CAS服务端的地址,这里不要加login-->  <param-value> https://sso.gdip.edu.cn/authserver</param-value>  </init-param>  <init-param>  <param-name>serverName</param-name>  <!--这里是应用地址，注意是域名:端口或者ip:端口-->  <param-value>http://app:8080</param-value>  </init-param>  <!--去身份认证的校验的时候也需要加一个编码-->  <init-param>  <param-name>encoding</param-name>  <param-value>UTF-8</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>CAS Validation Filter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- 该过滤器负责实现HttpServletRequest请求的包裹，比如允许开发者通过HttpServletRequest的getRemoteUser()方法获得SSO登录用户的登录名。 -->  <filter>  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>  <filter-class>org.jasig.cas.client.util.HttpServletRequestWrapperFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>CAS HttpServletRequest Wrapper Filter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- =============== 单点登录结束 ================= --> |

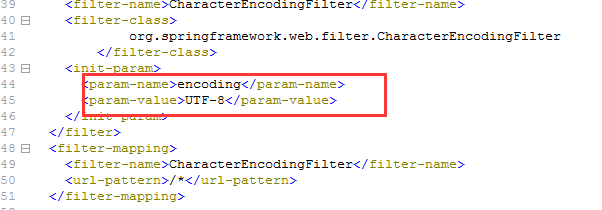
说明：

web项目的编码都encodeFilter的filter-mapping都是放到第一个的，这个执行顺序要注意的。Casclient客户端是不涉及到编码格式的，默认都是来自于项目的。

执行第一个getParameter()的时候，java将会按照编码分析所有的提交内容，而后续的getParameter()不再进行分析，所以setCharacterEncoding()无效。而对于GET方法提交表单是，提交的内容在URL中，一开始就已经按照编码分析提交内容，setCharacterEncoding()自然就无效。

所以请将如上的代码filter-mapping放到项目编码filter-mapping下面，保证最先执行项目编码过滤器。

注意：上面的配置要放在业务系统的web.xml文件中的编码下面，如下图所示部分是业务系统的项目编码。



#### 获取用户信息

第三方应用在配置好web.xml之后，需要调整代码用集成之后的方式获取当前登录用户。获取用户信息的示例代码如下：

|  |
| --- |
| //cas-client-3.2.1版本集成  String uid = request.getRemoteUser();  String cn = "";  Principal principal = request.getUserPrincipal();  if(principal!=null && principal instanceof AttributePrincipal){  AttributePrincipal aPrincipal = (AttributePrincipal)principal;  //获取用户信息中公开的Attributes部分  Map<String, Object> map = aPrincipal.getAttributes();  // 获取姓名,可以根据属性名称获取其他属性  cn = (String)map.get("cn");  } |

#### 集成应用退出

第三方应用在退出的时候需要重定向跳转到IDS的退出地址，具体示例代码如下(红色部分需要根据实际情况调整)，被集成业务系统退出，必须用调用身份认证的退出地址，并且要清除业务系统自身的cookie。

|  |
| --- |
| session.invalidate();  // ids的退出地址，ids6.wisedu.com为ids的域名 authserver为ids的上下文，logout为固定值  String casLogoutURL = " https://sso.gdip.edu.cn/authserver/logout";  // service后面带的参数为应用的访问地址，需要使用URLEncoder进行编码  String redirectURL=casLogoutURL+"?service="+URLEncoder.encode("http://app:8080/caslogin.jsp");  response.sendRedirect(redirectURL); |

#### 白名单（认证不拦截）

在第三方应用的web.xml中添加以下内容：（加色部分需要修改为实际参数），符合白名单规则的地址，不被身份认证拦截。白名单的配置仅支持正则表达式匹配。

|  |
| --- |
| <filter>  <filter-name>CAS Authentication Filter</filter-name>  <filter-class>org.jasig.cas.client.authentication.AuthenticationFilter</filter-class>  <init-param>  <param-name>casServerLoginUrl</param-name>  <param-value> https://sso.gdip.edu.cn/authserver/login</param-value>  </init-param>  <!---仅支持正则表达式匹配，支持英文逗号分隔，可以配置多个-->  <init-param>  <param-name>casWhiteUrl</param-name>  <param-value>/demo.\*/index.jsp</param-value>  </init-param>  <init-param>  <param-name>serverName</param-name>  <param-value>http://app:8090</param-value>  </init-param>  </filter> |

#### 认证接口工作过程

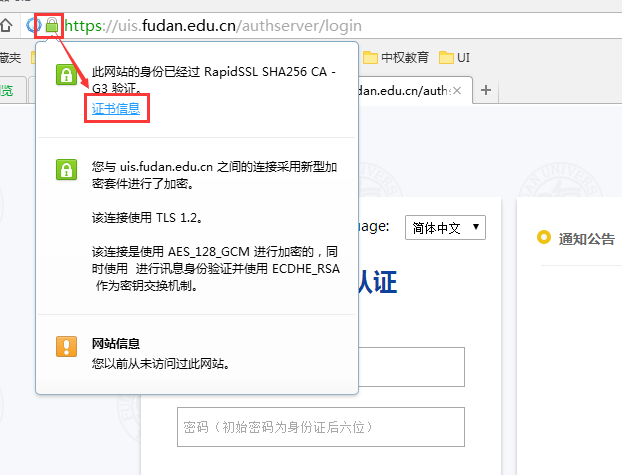


#### 集成应用添加SSL证书信任

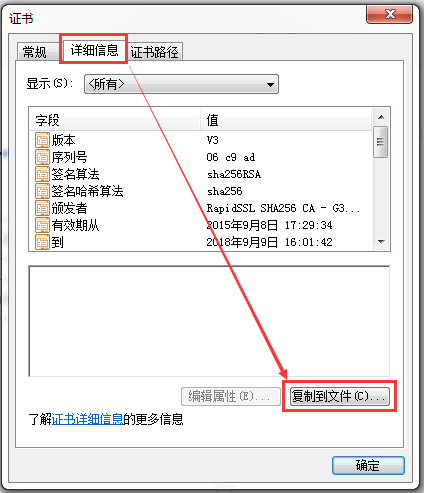
统一身份的集成，待集成的应用系统需要去统一身份平台验证令牌ticket，这个验证如果统一身份平台配置了https，则需要在第三方应用服务器导入SSL证书添加信任，一般情况下，java程序在该第三方应用所使用的jdk中把证书导入security\cacerts即可。

安装证书举例，360浏览器为例

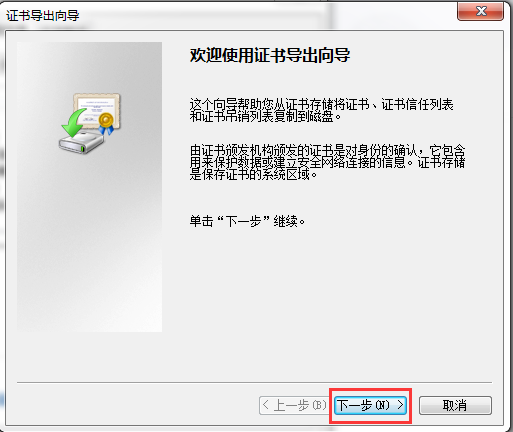
第一步：首先,需要用浏览器下载https证书，访问https://uis.fudan.edu.cn

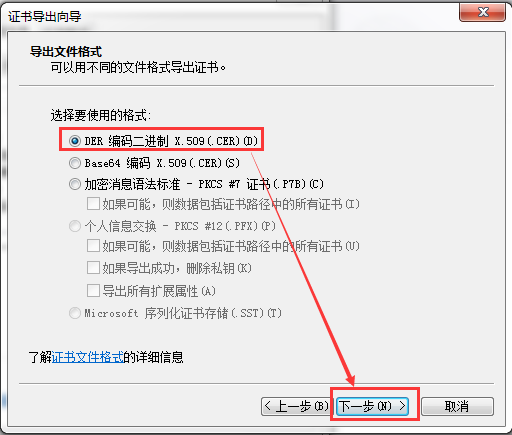


第二步：切换到证书的详细信息，并选择复制到文件

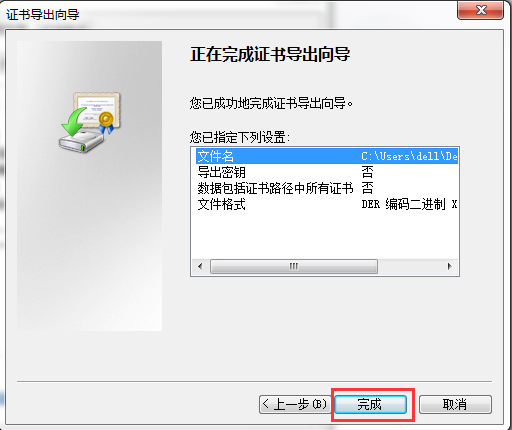


第三步：按证书导出向导，下一步选择证书格式，一般选择缺省的证书格式









第六步：复制证书到jdk环境下，例如：/opt/soft/fudan.cer

第七步：使用命令行 cd $JAVA\_HOME/security/cacerts

$JAVA\_HOME替换为你自己的JDK目录

|  |
| --- |
| /usr/java/jdk1.7.0\_80/jre/lib/security |

第八步：输入命令

|  |
| --- |
| keytool -import -alias cacerts -keystore cacerts -file /opt/soft/fudan.cer |

第九步:按提示输入cacerts证书库的密码，缺省为changeit

第十步:信任证书，导入完成。

第十一步:查看导入的证书，输入keytool -list -keystore cacerts

注: 更新证书前需输入keytool -delete -alias cacerts -keystore cacerts 先删除证书

## ASP集成

* 参考例子：asp.rar

### 认证接口集成说明

#### 新增登录入口

新增加应用访问入口，新增入口与CAS做集成，caslogin.asp

|  |
| --- |
| //CAS Server的登陆URL  var loginServer = "https://sso.gdip.edu.cn/authserver/login";  //CAS Server的验证URL  var validateServer = "https://sso.gdip.edu.cn/authserver/serviceValidate";  //当前集成系统所在的服务器和端口号，服务器可以是机器名、域名或ip，建议使用域名。端口不指定的话默认是80  //以及应用名称和新增加的集成登录入口  var loginasp = "http://app/aspsso/caslogin.asp";  //登录成功后重定向url参数  var strRedirecturl = Request.QueryString.Item("redirectUrl").Item;  if(strRedirecturl != null){  loginasp = loginasp + "?redirectUrl=" + strRedirecturl;  }  //判断是否已经登录, (需要集成系统开发人员完成)  If(Session(“uid”)){  Response.Redirect(strRedirecturl);  Response.End;  }else{  var ticket = Request.QueryString.Item("ticket").Item;  if (!ticket) {  var url = loginServer+"?service="+loginasp;  Response.Redirect(url);  Response.End;  } else {  //获取登录后的返回信息  var http = Server.CreateObject("Msxml2.ServerXMLHTTP.3.0");  //忽略https证书错误  http.setOption(2) = 13056;  var url = validateServer+"?ticket="+ticket+"&service="+loginasp;  http.open("GET",url,false);  http.send();  var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.4.0");  xmlDoc.async = false;  xmlDoc.loadXML(http.responseText);  //验证CAS登录是否成功  var successList = xmlDoc.getElementsByTagName("cas:authenticationSuccess");  if(successList.length == 1){  var ssouserList = xmlDoc.getElementsByTagName("cas:user");  //得到登录用户的账号userID  var userID = ssouserList.item(0).text;  var ssocnList = xmlDoc.getElementsByTagName("cas:cn");  var userName = ssocnList.item(0).text;  //这边如果需要取其他属性，也类似上面的姓名的取法  //实现集成系统的登录（需要集成系统开发人员完成）  //............实现代码...................  Session("uid")=userID;  Session("userName")=userName;  //实现登录完毕！  //如果登录成功，执行下面代码，否则按集成系统业务逻辑处理  var Rurl = Request.QueryString.Item("redirectUrl").Item;  if(Rurl == null){  //此处是集成系统的首页URL  Rurl = "http://app/aspsso/home.asp";  }  Response.Redirect(Rurl);  Response.End;  }  }  } |

上述例子是一种实现逻辑，访问受控网页跳转到caslogin.asp,如果登录了就直接回去了，如果没登录跳转到ids登录页登录完了就跳转回caslogin.asp,然后再跳转回一开始的受控网页，登录完成。

#### 修改原有登出

(可选，如果不需要单点登出，这一部分可以不做)

在原有登出逻辑的基础上，redirect到统一身份认证logout链接，如下(跳转到哪一页由集成系统自己决定)

[https://am.wisedu.com/authserver/logout?service={encoded](https://am.wisedu.com/authserver/logout?service=%7bencoded)(集成系统首页URL)}

|  |
| --- |
| Session.Abandon();  var logoutURL = <https://sso.gdip.edu.cn/authserver/logout>"+  "?service=http://app/aspsso/home.asp";  Response.Redirect(logoutURL);  Response.End; |

登出现在有2点没做

其他单点登出导致的当前应用登出，如果有需要可以提供java方式的逻辑。

当前集成应用超时处理，跟上面的登出链接类似处理。(需要监听应用会话)

**【注意】：做这部分实现的是，我这边退出，别人应用也退出；但是别人退出，我这边不退出。**

## donet集成

* 参考例子：cas\_demo.zip

### .net认证接口集成说明

代理认证配置完后，均需至统一身份管理平台授权访问。先登录身份认证管理平台，在认证管理的认证应用模板，添加需代理认证的应用；添加完应用后，需给相应的组或者帐号授权，允许其访问该应用。

#### 认证接口工作过程



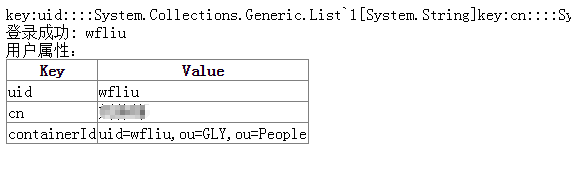
#### 拷贝提供的dll文件

将“…/cas\_demo/ExampleWebSite/Bin”下DotNetCasClient.dll文件拷贝到客户端应用所在bin目录下，项目添加该dll的引用。

#### 用户属性获取代码参考

代码源文件地址..\cas\_demo\ExampleWebSite\Userinfo.aspx.cs

查看用户信息地址：http://demo访问IP地址/Userinfo.aspx



|  |
| --- |
| public partial class Userinfo : System.Web.UI.Page  {  protected void Page\_Load(object sender, EventArgs e)  {  string userName = User.Identity.Name ;    CasPrincipal p =Page.User as CasPrincipal ;    //获得所有传递过来的用户属性  Dictionary<string, string> userAttrs = new Dictionary<string, string>();  var cnValue=UserAttrHelper.GetUserAttrValueByParamName("cn"); //姓名  var aliasValue=UserAttrHelper.GetUserAttrValueByParamName("alias"); //别名  userAttrs.Add("cn",string.Join(",",cnValue));  userAttrs.Add("alias", string.Join(",", aliasValue));    user.Text = userName.ToString();  attr.DataSource = userAttrs;  attr.DataBind();  }  } |

#### 修改web.config文件

修改…\cas\_demo\ExampleWebSite\web.config；casClientConfig里设置环境信息，httpmodules为必选，否则不能实现单点登录。

|  |
| --- |
| <?xml version="1.0"?>  <configuration>  <configSections>  <section name="casClientConfig" type="DotNetCasClient.Configuration.CasClientConfiguration, DotNetCasClient"/>  <section name="FilterUrls" type="DotNetCasClient.Configuration.ReWritingConfigurationSetHandler, DotNetCasClient"/>  </configSections>  <appSettings/>  <connectionStrings/>  <casClientConfig casServerLoginUrl="https://sso.gdip.edu.cn/authserver/login" casServerUrlPrefix="https://sso.gdip.edu.cn/authserver/" serverName="http://oa.xxx.edu.cn" notAuthorizedUrl="~/NotAuthorized.aspx" cookiesRequiredUrl="~/CookiesRequired.aspx" redirectAfterValidation="true" gateway="false" renew="false" singleSignOut="true" ticketTimeTolerance="5000" ticketValidatorName="Cas20" serviceTicketManager="CacheServiceTicketManager" gatewayStatusCookieName="CasGatewayStatus"/>  <system.web>  <authentication mode="Forms">  <!--身份认证系统地址-->  <forms loginUrl="https://sso.gdip.edu.cn/authserver/login" timeout="30" defaultUrl="~/Default.aspx" cookieless="UseCookies" slidingExpiration="true" path="/"/>  </authentication>  <authorization>  <allow users="\*"/>  </authorization>  <roleManager enabled="true" defaultProvider="AspNetReadOnlyXmlRoleProvider">  <providers>  <add name="AspNetReadOnlyXmlRoleProvider" type="DotNetCasClient.Security.ReadOnlyXmlRoleProvider" xmlFileName="~/App\_Data/UserRoles.xml"/>  </providers>  </roleManager>  <siteMap enabled="true">  <providers>  <clear/>  <add siteMapFile="Web.sitemap" name="AspNetXmlSiteMapProvider" type="System.Web.XmlSiteMapProvider"/>  </providers>  </siteMap>  <compilation debug="true" targetFramework="4.0">  </compilation>  <pages controlRenderingCompatibilityVersion="3.5" clientIDMode="AutoID">  <controls>  <add tagPrefix="cas" tagName="CookieViewer" src="~/Controls/CookieViewer.ascx"/>  <add tagPrefix="cas" tagName="TicketManager" src="~/Controls/TicketManager.ascx"/>  </controls>  </pages>  <httpModules>  <add name="DotNetCasClient" type="DotNetCasClient.CasAuthenticationModule,DotNetCasClient"/>  </httpModules>  </system.web>  <!--编写自定义的url过滤器，符合此过滤器的页面 地址不需要登录，也能访问-->  <FilterUrls>  <Rules>  <Url>  <ServletName>Test</ServletName>  <Url-Mapping>^/cas/test\_(/S\*)(.aspx)</Url-Mapping>  </Url>  </Rules>  </FilterUrls>  <system.diagnostics>  <trace autoflush="true" useGlobalLock="false"/>  <sharedListeners>  <add name="TraceFile" type="System.Diagnostics.TextWriterTraceListener" initializeData="C:\inetpub\DotNetCasClient.Log" traceOutputOptions="DateTime"/>  </sharedListeners>  <sources>  <source name="DotNetCasClient.Config" switchName="Config" switchType="System.Diagnostics.SourceSwitch">  <listeners>  <add name="TraceFile"/>  </listeners>  </source>  <source name="DotNetCasClient.HttpModule" switchName="HttpModule" switchType="System.Diagnostics.SourceSwitch">  <listeners>  <add name="TraceFile"/>  </listeners>  </source>  <source name="DotNetCasClient.Protocol" switchName="Protocol" switchType="System.Diagnostics.SourceSwitch">  <listeners>  <add name="TraceFile"/>  </listeners>  </source>  <source name="DotNetCasClient.Security" switchName="Security" switchType="System.Diagnostics.SourceSwitch">  <listeners>  <add name="TraceFile"/>  </listeners>  </source>  </sources>  <switches>  <add name="Config" value="Information"/>  <add name="HttpModule" value="Verbose"/>  <add name="Protocol" value="Information"/>  <add name="Security" value="Information"/>  </switches>  </system.diagnostics>  </configuration> |

## PHP集成

* 参考例子：php.rar

### 认证接口集成

代理认证配置完后，均需至统一身份管理平台授权访问。先登录身份认证管理平台，在认证管理的认证应用模板，添加需代理认证的应用；添加完应用后，需给相应的组或者帐号授权，允许其访问该应用。

### PHP客户端集成说明

<https://wiki.jasig.org/display/CASC/phpCAS+examples#phpCASexamples-AsimpleCASclient>

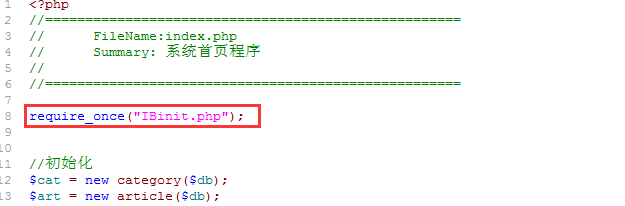
官方参考文档

#### 拷贝开发包并注册

Php开发包比较简单。CAS-1.3.2.tgz为官方client内置大量的example。我们也提供了一个简单demo。可以查看demo/目录下的文件。

在网站的公有php文件中（index.php），添加一下代码，并将CAS文件夹放到第三方应用中的同级目录下。（就是要保证引用到CAS文件夹下的CAS.php）

例如，某个php写的网站，打开index.php，在下图红色框下面添加下面的代码。



|  |
| --- |
| require\_once dirname(\_\_FILE\_\_)."/CAS/CAS.php";  //指定log文件  phpCAS::setDebug('./log.log');  //指定cas地址,第一个为cas版本，一般为CAS\_VERSION\_2\_0  第二个参数为hostname，第三个为ids的端口，第四个是ids的上下文，第五个是是否是https。  phpCAS::client(CAS\_VERSION\_2\_0,'idp.test.com',8443,'cas',true);  //sso退出时，cas会请post应用带上logoutRequest参数，请求应用地址。  //设置no ssl，即忽略证书检查。如果需要ssl，请用 phpCAS::setCasServerCACert()设置ssl证书。  phpCAS::setNoCasServerValidation();  //phpCAS::handleLogoutRequests()可以响应sso退出请求，注销当前用户认证凭据。  phpCAS::handleLogoutRequests();  phpCAS::forceAuthentication();  //$\_REQUEST['logout'] 只是本应用退出参数。同时退出所有sso应用。  //param可以指定 业务系统的service和url。可以参考cas官方说明。  if(isset($\_REQUEST['logout'])){  $param = array('service'=>'http://demo.cas.wisedu.cn:3273/');  phpCAS::logout($param);  exit;  }  用户信息可由：  phpCAS::getUser();  phpCAS::getAttributes();  两个方法获取 |

#### 认证接口工作过程

统一认证头流程

#### 认证接口集成

接口返回的时候就已经返回用户信息。可参考上述描述。