**IDStarV5.0应用集成接口说明**

**JAVA**

|  |  |
| --- | --- |
| **编制人员：** | **许继锋** |
| **编制部门：** | **平台技术部** |

|  |  |  |
| --- | --- | --- |
| 审核确认 | | |
| **金智教育** |  |  |
| 签字：  日期： | 签字：  日期： | 签字：  日期： |

修改记录表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序 | 修改人 | 修改内容 | 批准人 | 生效日期 | 版本号 |
| 1 | 许继锋 | 创建文档 |  |  | V0.1 |
| 2 | 衡南南 | 完善文档 |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |
| 10 |  |  |  |  |  |

目录

[1 前言 4](#_Toc350931006)

[1.1 编写目的 4](#_Toc350931007)

[1.2 服务对象 4](#_Toc350931008)

[2 集成方式概述 5](#_Toc350931009)

[3 认证接口集成 7](#_Toc350931010)

[3.1.1 Java客户端部署 7](#_Toc350931011)

[3.1.2 .NET客户端部署 9](#_Toc350931012)

[3.1.3 PHP客户端部署 10](#_Toc350931013)

[3.2 认证接口集成过程 12](#_Toc350931014)

[3.2.1 认证接口工作过程 12](#_Toc350931015)

[3.2.2 认证接口集成 12](#_Toc350931016)

[4 LDAP接口集成 13](#_Toc350931017)

# 前言

## 编写目的

本文档作为IDStarV5.0版本中应用系统的认证集成手册，旨在指导校内应用系统实现校内统一身份认证和单点登录集成。

## 服务对象

|  |  |
| --- | --- |
| **读者对象** | **阅读范围** |
| 产品开发项目组人员 | 能熟悉掌握整个产品的流程和实施方法，能做到顺利完成产品项目或指导产品项目的实施。 |
| 产品维护人员 | 阅读全部内容 |
| 工程项目经理 | 阅读全部内容 |
| 产品部署实施人员 | 阅读全部内容 |

# 集成方式概述

将高校的各个业务系统与身份管理平台连接，通过身份管理平台实现用户身份认证和单点登录功能，这个过程即应用系统与身份管理平台的集成。IDStarV5.0主要包括以下集成方式：

* 认证接口：这是各个应用系统与身份管理平台集成的最主要的方式。各应用将认证接口的客户端开发包集成在各应用之中，替换自身原有独立的身份认证功能，通过身份管理平台实现身份认证和单点登录过程。该接口目前涵盖的类型包括JAVA、 PHP和 .NET(2.0+)，适合于不同语言和平台的应用程序。
* LDAP接口：对于选课选这样的高并发应用，我们提供LDAP接口，以满足认证的性能需求。该接口直接通过LDAP向应用系统提供认证服务，但是牺牲了单点登录功能。



# 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。

### Java客户端部署

#### 拷贝开发包

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WebContent\WEB-INF\web.xml文件

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>

<!-- 该过滤器用于实现单点登出功能，可选配置。 -->

<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>

<param-value> http://ids.cumt.edu.cn/authserver/serviceValidate</param-value>

<!--这里的server是服务端的地址-->

</init-param>

<init-param>

<param-name>serverName</param-name>

<param-value>人事系统域名 </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>

<param-value> http://ids.cumt.edu.cn/authserver</param-value>

</init-param>

<init-param>

<param-name>serverName</param-name>

<param-value>人事系统域名</param-value>

</init-param>

<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，可选配置。 -->

<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>

<!-- 该过滤器使得开发者可以通过org.jasig.cas.client.util.AssertionHolder

来获取用户的登录名。

比如AssertionHolder.getAssertion().getPrincipal().getName()。 -->

<filter>

<filter-name>CAS Assertion Thread Local Filter</filter-name>

<filter-class>org.jasig.cas.client.util.AssertionThreadLocalFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>CAS Assertion Thread Local Filter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<!-- =============== 单点登录结束 ================= -->

备注：

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，保证最先执行编码过滤器。

#### 获取用户信息代码范例

通过request获取：

String uid = request.getRemoteUser();//获取登录用户id

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();

Iterator<String> it = map.keySet().iterator();

while (it.hasNext()) {

String k = it.next();

response.getWriter().printf("%s:%s\r\n", k, map.get(k));

}

}

通过org.jasig.cas.client.util.AssertionHolder获取：

Assertion assertion = AssertionHolder.getAssertion();

AssertionHolder.getAssertion().getPrincipal()

获取了attributePrinciple对象，就可以取到属性值

## 认证接口集成过程

### 认证接口工作过程



### 认证接口集成

1. 接口返回的时候就已经返回用户信息。可参考上述描述。