**Apache Shiro入门**

Apache Shiro专注于易用性，因此您可以很轻松的使用它提供的安全，稳定的身份验证，授权，加密和会话管理等功能。通过Shiro提供的易于理解的API，您可以快速轻松地保护任何应用程序。下面就开始吧！

**基础**

**Apache Shiro Java安全框架简介**

Apache Shiro功能的高级概述。

**架构**

Apache Shiro的架构概述。

**术语**

常见安全概念和关注点的定义列表。

**Apache Shiro项目背景**

就像大多数有用的工具一样，Apache Shiro是出于实际需要而创建的。详细了解项目历史和项目所肩负的使命。Learn more about the project history and mission statement

**教程**

**10分钟教程**

在10分钟内了解Shiro 框架的所有细节。这个简单的教程快速地展示了开发人员如何在他们的应用程序中使用Shiro。

**你的第一个Shiro应用程序**

如果您刚接触Apache Shiro，这个简短的教程将向您展示如何创建一个简单的应用程序，它是由Apache Shiro来提供保护的。在创建的过程中，我们将介绍Shiro的核心概念，以帮助您熟悉Shiro的设计和API。

**其他资源**

**介绍性文章......以及其他的！**

由Apache Shiro社区成员编写的文章和指南。

**Apache Shiro Java安全框架简介**

**什么是Apache Shiro？**

Apache Shiro是一个功能强大且灵活的开源安全框架，它可以很好地处理身份验证，授权，企业会话管理和加密。

Apache Shiro的首要目标是易于使用和理解。有时保证系统的安全可能非常复杂，甚至是痛苦的，但是这些问题是可以解决的。框架应尽可能掩盖那些复杂的东西，并提供简洁直观的API，使开发人员能轻松地给应用程序提供安全保证。

以下是Apache Shiro可以做的一些事情：

检察确认用户身份

控制用户访问，例如

确认是否为用户分配了某个安全角色

确定是否允许用户执行某些操作

在任何环境中使用Session API，即使没有Web容器或EJB容器也是如此

响应在身份验证，访问控制过程中或者是session生命周期期内发生的事件

将一个或者多个用户安全数据聚集起来，用一个“综合视图”来展示

启用单点登录（SSO）功能

无需登录即可为相关用户启用“记住我”服务

还有更多 - 所有这些都集成在一个易于使用的API中。

Shiro尽可能的为所有应用程序实现这些目标 - 从最简单的命令行应用程序到最大的企业应用程序都可以使用shiro，并且不强制依赖其他第三方框架，容器或应用程序服务器。当然，该项目旨在尽可能地融入这些应用程序中。它在任何应用程序中都是开箱即用。

Apache Shiro功能

Apache Shiro是一个具有许多功能的综合的应用程序安全框架。下图展示了Shiro集中精力做的东西，并且此参考手册将以类似方式组织：

shiro开发团队提出“保证应用程序安全的四大基石是身份验证，授权，会话管理和加密” 。Shiro项目的目标就是打造可以保证应用程序安全的“四大基石”。

身份验证：有时称为“登录”，这是证明此用户是他自己所说的身份的行为。

授权：也称为访问控制，即决定“谁”可以访问“什么”。

会话管理：管理特定用户的会话，即使在非Web或EJB应用程序中也可以管理特定用户的会话

加密：使用加密算法保证数据安全，并且保持数据的易用性。

shiro还有其他功能，它们支持和强化各种应用程序中的这些关键点，尤其是：

Web支持：Shiro的Web API可以轻松保护Web应用程序。

缓存：缓存是Apache Shiro API中的一级公民，它可以确保安全操作快速高效。

并发：Apache Shiro的并发功能支持多线程的应用程序。

测试：测试支持可以帮助您编写单元测试和集成测试，确保您的代码得到预期的保护。

“以某个身份运行”：在特定条件下，允许某一用户借用其他用户的身份。有时在管理中，这种借用身份的行为很有用。

“记住我”：记住各个会话中用户的身份，这样他们只需要在必要时登录。

[Apache Shiro](http://shiro.apache.org/architecture.html" \l "apache-shiro-architecture)架构

Apache Shiro的设计目标是通过shiro提供的直观、易用的功能，来轻松的保证应用程序的安全性。Shiro的核心设计考虑了大多数人在与应用程序交互时对应用程序安全性的看法。

应用程序通常根据“用户故事”设计。也就是说，您通常会（或应该）根据用户与软件交互的方式设计用户界面或服务API。例如，您可能会说，“如果用户是在登录状态与我的应用程序交互，则会向他们显示一个按钮，他们可以单击这个按钮查看其他们的帐户信息。如果他们没有登录，我会显示一个登录按钮。“

上面的例子说明应用程序主要是为满足用户要求和需求而编写的。即使“用户”是另一个软件系统而不是人类，您仍然会根据当前与您的软件交互的人（或物）的身份，编写代码以表现他们的身份所应有的行为。reflect behavior

Shiro在自己的设计中体现了这些概念。通过匹配软件开发人员已经很熟悉的事物，几乎在任何应用程序中都可以很轻松的使用Apache Shiro。

High-Level Overview高层次概述

在最高的概念层面，Shiro的架构有3个主要概念：主题Subject，安全管家SecurityManager 和领域Realms。下图是这些组件如何交互的概述，我们将在下面介绍每个概念：

主题：正如我们在之前提到的，主题本质上是按照系统安全性要求，当前用户所能看到的特定“视图”。a security specific ‘view’ of the the currently executing user

鉴于“用户”这个词通常意味着一个人，因此主题可以是一个人，也可以是第三方服务，守后台帐户，计划任务或任何类似的东西。也就是说当前正在与软件交互的任何东西都是主题Subject。

**安全管家SecurityManager：**安全管家是Shiro架构的核心，它充当一种“伞形”对象，和其内部的其他安全组件，共同形成一个对象图。但是，一旦配置了应用程序的安全管家（SecurityManager）及它的内部对象图，安全管家就会被放在一边不管了。应用程序开发人员几乎将所有时间花在主题的API上。

我们稍后会详细讨论SecurityManager，但重要的是要意识到，当你与主题（Subject）交互时，实际上是SecurityManager在幕后完成了，进行任何Subject安全操作的所需要做的繁重任务。上面的基本流程图中很好的说明了这一点。

**领域Realms** ：领域充当Shiro与应用程序安全数据之间的“桥梁”或“连接器”。当我们真正与安全相关的数据（如用户帐户）进行交互——以执行身份验证（登录）和授权（访问控制）时，Shiro会从为应用程序配置的一个或多个领域中查找许多这些内容。

从这个意义上讲，Realm本质上是一个着眼于安全性的DAO：它封装了数据源的连接细节，并且可以给shiro提供它需要的相关数据。配置Shiro时，必须至少指定一个Realm用于身份验证和/或授权。SecurityManager至少需要一个领域，也可以配置多个领域。

Shiro提供了开箱即用的Realms。通过shiro提供的realms我们可以连接到许多安全数据源（也称为目录），如LDAP，关系数据库（JDBC），文本配置源（如INI）和属性文件等。如果默认的Realms 不能满足您的需要，您可以插入自己的Realm实现来表示自定义数据源。

与其他内部组件一样，Shiro的 SecurityManager管理如何使用Realms获取Subject实例的安全性数据和身份数据。安全性数据和身份数据经常被描述为主题实例。

[Detailed Architecture](http://shiro.apache.org/architecture.html" \l "detailed-architecture)架构详情

下图显示了Shiro架构的核心概念，它们的后面都有一个简短的介绍：

主题

当前与软件交互的实体（用户，第三方服务，定时作业等）的特定的安全性“视图”。

安全管理器

如上所述，安全管理器是Shiro架构的核心。它就像是一个“伞形”对象，它协调其管理的组件，以确保它们一起平稳运行。它还管理Shiro对每个用户提供的视图，因此它知道如何对每个用户执行安全操作。

认证

认证是指负责执行和响应用户的身份验证（登录）请求的组件。当用户尝试登录时，该逻辑由Authenticator执行。Authenticator知道如何与一个或多个领域进行协调，在这些领域中存储着相关用户/帐户的信息。从这些领域中获得的数据是用来验证用户的身份，以保证用户确实是他们自己所说的那样。

身份验证策略

如果配置了多个Realm，身份验证策略将协调这些领域，以确定在什么情况下身份验证的尝试是成功的还是失败（例如，如果一个领域成功但其他领域失败，尝试是否成功？必须所有领域成功吗？或者是只需要第一个成功就可以了？

授权人

授权人是负责确定应用程序中用户访问控制的组件。也就是说这种机制最终判定用户是否可以做某事。与证明人一样，授权人也知道如何协调多个后端数据源，以获取角色和权限信息。授权人使用获得的信息来确定是否允许用户执行某些指定操作。

会话管理器

会话管理器知道如何创建和管理用户会话生命周期，以便为所有环境中的用户提供强大的会话体验。Shiro拥有安全框架领域一项独一无二的功能 - 即使没有可用的Web / Servlet或EJB容器，Shiro也能够在任何环境中本地自然地管理用户Sessions。默认情况下，Shiro将使用现有的会话机制（例如Servlet容器），但如果没有，例如在独立应用程序或非Web环境中，它将使用其内置的企业会话管理，来给开发人员提供相同的编程体验。SessionDAO的存在是为了允许任何数据源用于持久化会话。

SessionDAO

SessionDAO代表SessionManager执行会话持久性（增删改查）操作。这允许将任何数据块插入会话管理的基础结构中。

缓存管理

CacheManager创建和管理Cache实例（提供给shiro的其他组件使用）生命周期。由于Shiro可以访问许多后端数据源以进行身份验证，授权和会话管理，因此缓存一直是框架中一个非常好的功能，它可以提高系统的性能——减少访问后端数据源时时间所需要的时间。任何开源或企业级缓存产品都可以插入Shiro，以提供快速高效的用户体验。

加密

加密技术是对企业安全框架的自然补充。Shiro的加密包包含易于使用和理解的秘密暗号，哈希（也称为摘要）和不同的编解码器的实现。该包中的所有类都经过精心设计，易于理解和使用。使用Java原生加密支持的人都知道，它是一个具有难以驯服的动物。Shiro的加密API简化了复杂的Java机制，使普通人也能轻松的使用。

领域

如上所述，领域充当Shiro与应用程序安全数据之间的“桥接”或“连接器”。当实际同系统的安全数据（如用户帐户）进行交互，以执行身份验证（登录）和授权（访问控制）时，Shiro会从为应用程序配置的一个或多个领域中查找相应的内容。您可以根据需要配置任意数量的域（通常一个数据源一个领域），Shiro将根据身份验证和授权的需要，来协调多个领域共同工作。