VIVO V1.5 安装指南

目录

这份文件是 VIVO 安装过程的总结。如需有关此版本的详细信息，请咨询的 VIVO V1.5 发布公告。这和其他文档支持页面上可以找到在 VIVOweb.org。

这些说明假设您正在执行一个干净的安装，包括一个现有的数据库，VIVO 的主目录，并删除先前安装 Tomcat 的 webapps 目录。如果你安装在现有安装的早期版本，产品功能可能无法达到预期。

VIVO 分布目录

这是当您解压开包装 VIVO 分布文件（见第三步，下同）。在这里您将创建您的 deploy.properties 文件（见第四步，下面），并在那里你将 VIVO 主题或代码进行任何修改。

Tomcat 内部署 VIVO

当你运行构建脚本编译和部署 VIVO（见第六步，下同），文件将被部署到 Tomcat 的目录里面。这是实际的执行代码 VIVO，但你不会需要看它或改变它。如果您需要更改 VIVO，分发目录中进行更改，并再次运行构建脚本。告诉构建脚本的 Tomcat 在哪里可以找到设置 tomcat.home 在 deploy.properties 文件（见第四步，下面）。

VIVO 的目录

VIVO 将使用该区域来存储一些数据。上传的图像文件都存储在这里，Solr 的主目录中也存放在这里。无论您选择，您可以创建此。告诉 VIVO 在哪里可以找到主目录设置 vitro.home.directory 在 deploy.properties 文件（见第四步，下面）。 VIVO 开始前，您必须创建这个目录，你必须确保 Tomcat 有权限读取和写入到这个目录中，运行时。

MySQL 数据库

从本质上讲，你 MySQL 将给予 VIVO 的存储的所有数据。这个数据的实际位置取决于你用什么样的系统，你如何安装 MySQL（请参阅下面的步骤）。您将要访问的数据通过 VIVO，或偶尔通过 MySQL 客户端应用程序。

安装步骤

1。安装所需的软件
2。创建一个空的 MySQL 数据库
3。下载的 VIVO 应用程序源
4。指定部署

5。编译并部署
6。配置 Tomcat
7。启动 Tomcat
8。登录并添加 RDF 数据
9。设置联系人的电子邮件地址（如果使用“联系我们”的形式）
10。安装 Apache Tomcat 的连接器
11。使用外部认证系统与 VIVO

一，安装所需的软件

安装 VIVO 之前，确保所需的计算机上安装以下软件：

o Java (SE) 1.6.x http://java.sun.com
o VIVO does not work with OpenJDK
o VIVO does not work with Java 1.7
• Apache Tomcat 6.x or 7.x http://tomcat.apache.org
• Apache Ant 1.8 or higher, http://ant.apache.org
• MySQL 5.1 or higher, http://www.mysql.com

一定要设置环境变量 JAVA_HOME 和 ANT_HOME 可执行文件，并添加到您的路径操作系统和软件支持网站的安装方向。

此版本支持以下浏览器

· Mac:

o Chrome 8.0.552.237
o FireFox 3.6.13
o Opera 10.6.2 o Safari 5.0.3

• PC:

o Chrome 8.0.552.273
o FireFox 10.0.2
o Internet Explorer 7, 8, 9
o Opera 10.6.2

二。创建一个空的 MySQL 数据库

确定数据库名，用户名和密码。登录到你的 MySQL 服务器和 MySQL 中创建一个新的数据库，使用 UTF-8 编码。您将需要这些值第四步，当你配置的部署属性。在 MySQL 命令行中，DBNAME，用户名和密码，你在 SQL 文本中使用这些命令，您可以创建数据库和用户。

CREATE DATABASE dbname CHARACTER SET utf8;

授予数据库用户的访问。例如：

GRANT ALL ON dbname.* TO 'username'@'hostname' IDENTIFIED BY 'password';

跟踪的数据库名，用户名和密码第四步。

三。下载的 VIVO 应用源

VIVO 应用程序源 1.5.zip 或 REL-1.5.gz 文件下载和解压您的 Web 服务器上：http://vivoweb.org/download

四。指定部署属性

在 VIVO 分布目录的顶层，复制文件 example.deploy.properties，到一个文件名为简单 deploy.properties 的。该文件将用于编译和部署的几个属性。

视窗：对于那些上安装 Windows 操作系统，包括 Windows 驱动器，并使用正斜杠“/”，而不是背面斜线“\”目录中的位置，如 C:/ tomcat 的。

外部身份验证：如果你想使用像 Shibboleth 的 CUWebAuth 的外部验证系统，在这个文件中，你将需要设置两个附加属性。请参阅下面的步骤有权使用 VIVO 外部认证系统。

项目名字	示例值
默认命名空间：VIVO 安装 RDF 资源可以使用链接数据的收获。 RDF 资源的 URI 请求重定向到客户指定的 HTML 或 RDF 陈述。为了做到这一点，VIVO 的默认命名空间必须有一定的结构，并开始公开网址 VIVO 安装。例如，如果网络地址的 VIVO 安装是“http://vivo.example.edu/”默认的命名空间必须设置为“http://vivo.example.edu/individual/”，以支持链接的数据。同样，如果的 VIVO 安装在“http://www.example.edu/vivo”默认的命名空间，必须设置为“http://www.example.edu/vivo/individual/” *命名空间必须以“individual/”（包括结尾的斜线）	
Vitro.defaultNamespace	http://vivo.mydomain.edu/individual/
位于目录下项目外代码。在大多数部署，这是设置的/体外核心的（此设置指向在开发环境中的其他地方并不少见）。	
vitro.core.dir	./vitro-core
安装 Tomcat 的目录。	
tomcat.home	/usr/local/tomca
VIVO 应用程序的名称。	
webapp.name	vivo
Solr 的上下文中使用本地 VIVO 搜索的 URL。应包括：scheme + servername + port + vivo_webapp_name + "solr"在标准安装，Solr 的上下文将作为 VIVO，并在相同的 Tomcat 实例在同一服务器上。路径将是的 VIVO webapp.name（上面指定的）+"solr"	
vitro.local.solr.url	http://localhost:8080/vivosolr
限制访问 Solr 的搜索平台。一个或多个正则表达式，用逗号隔开。当一个请求到 Solr，请求者的 IP 地址必须匹配的模式之一，将被拒绝或请求。示例： • vitro.local.solr.ipaddress.mask = 127\.0\.0\.1 • vitro.local.solr.ipaddress.mask=127\.0\.0\.1,0:0:0:0:0:0:0:1 • vitro.local.solr.ipaddress.mask = 169.254.*	

- vitro.local.solr.ipaddress.mask = 127\.0\.0\.1
- vitro.local.solr.ipaddress.mask = 127\.0\.0\.1,0:0:0:0:0:0:0:1
- vitro.local.solr.ipaddress.mask = 169.254.*

vitro.local.solr.ipaddress.mask	127\.0\.0\.1|[0:]+:1

目录对 VIVO 应用程序将存储数据，它造成的上传的文件（通常是图像）和 Solr 的搜索索引。要确保此目录存在并且是可写的用户运行 Tomcat 服务。

vitro.home.directory	/usr/local/vivo/data

指定的 SMTP 主机，应用程序将用于发送电子邮件（可选）。如果这是空白，接触的形式将被隐藏和禁用，用户将不会收到通知到他们的帐户的变化。

email.smtpHost	smtp.servername.edu

指定的电子邮件地址作为发件人的电子邮件通知用户（可选）将出现。如果用户回复的通知，该地址将收到答复。如果用户的 E-mail 地址是无效的，这个地址将收到错误通知。如果这是空白，用户将无法更改帐户的通知。

email.replyTo	vivoAdmin@my.domain.edu

指定你的数据库的 JDBC URL。更改结尾的网址，以反映你的数据库名称（如果它不是“VIVO”）。

VitroConnection.DataSource.url	jdbc:mysql://localhost/vivo

更改用户名相匹配的授权用户，您在 MySQL 中创建。

VitroConnection.DataSource.username	Username（数据库用户名）
VitroConnection.DataSource.password	Password（数据库密码）

指定活动连接的最大数目的数据库连接池，以支持预期数量的并发页面请求。这是没有必要调整此值时使用的 RDB 组态。

VitroConnection.DataSource.pool.maxActive	40

指定将被允许在连接池中保持空闲的数据库连接的最大数目。默认为活动连接的最大数量的 25%。

VitroConnection.DataSource.pool.maxIdle	10

更改的的 DBTYPE 设置使用一个数据库 MySQL 之外。否则这个值不变。值可能是 DB2, derby, HSQLDB, H2, MySQL, Oracle, PostgreSQL, 和 SQLServer. 到 http://openjena.org/wiki/SDB/Databases_Supported 了解更多信息。

VitroConnection.DataSource.dbtype	MySQL

指定一个驱动程序类名使用一个数据库 MySQL 之外。否则这个值不变。该驱动程序必须加入此 JAR 文件的 webapp/ lib 目录上述指定 vitro.core.dir 内的。

VitroConnection.DataSource.driver	com.mysql.jdbc.Driver

更改验证查询用来测试数据库连接，只有在必要时才使用数据库 MySQL 之外。否则，离开这个值不变。

VitroConnection.DataSource.validationQuery	SELECT 1
指定对 VIVO 应用程序的根用户帐户的电子邮件地址。此用户将有一个初步的临时密码“rootPassword'。系统将提示您创建一个新的密码，在第一次登录。 注意：root 用户帐户访问所有的数据和在 VIVO 的所有操作。作为根用户身份登录时，可能会令人吃惊的数据视图。最好的方法是创建一个站点管理员帐户，用于每天的行政任务。	
rootUser.emailAddress	vivoAdmin@my.domain.edu
URI 的一个属性，可以用来与用户帐户相关联的个人。当用户登录一个名字，此属性的值相匹配，用户将被授权编辑，个人（属性的值必须是一个字符串或无类型文本）。 ＃networkId 的 selfEditing.idMatchingProperty http://vivo.mydomain.edu/ns 如果外部验证系统要使用像 Shibboleth 的 CUWebAuth 的是，这些属性怎么说，应标明“登录”按钮，将包含 HTTP 头的认证系统中的用户 ID。如果这样的系统是不被使用，离开这些注释掉。咨询更多详细的安装说明。	
externalAuth.buttonText Log in using BearCat Shibboleth	externalAuth.netIdHeaderName remote_userID
VIVO V1.4（及更高版本）的缓存机制的方式缓解这个问题，因此我们可以放心地设置默认情况下，启用此项功能。	
visualization.temporal	enabled
默认情况下，应用程序将尝试尽最大的猜测，在顶层组织在实例。如果你不满意这个选择，取消了将它设置为你要识别的顶层组织的 URI 组织个人。它将被用来作为默认的图形可视化的时间时，通过一个明确的组织结构不被呈现。例如，要使用“Ponce School of Medicine”作为顶级组织：visualization.topLevelOrg= http://vivo.psm.edu/individual/n2862	
visualization.topLevelOrg	http://vivo-trunk.indiana.edu/individual/topLevelOrgURI
绝对文件路径，指向实用程序的根目录。您必须包括最后的斜杠。	
harvester.location	/usr/local/vivo/harvester/
个别的类型，我们可以创建代理编辑。如果被省略，默认为 http://www.w3.org/2002/07/owl#Thing	
proxy.eligibleTypeList	http://xmlns.com/foaf/0.1/Person, http://xmlns.com/foaf/0.1/Organization
只显示最合适的数据值基于浏览器所提供的 Accept-Language 头。如果没有设置，默认值是 true。	
RDFService.languageFilter	true
这些值用于部署作为一个 OpenSocial 容器 VIVO 结合 OpenSocial 小工具（见第十二步，下同）。如果您正在创建一个 VIVO 安装不使用 OpenSocial 小工具，这些值被省略。	

OpenSocial.shindigURL OpenSocial.tokenService OpenSocial.tokenKeyFile OpenSocial.sandbox	

五，编译和部署

在命令行中，从 VIVO 分布的顶层目录，键入：

Ant all

建立 VIVO 和部署到 Tomcat 的 webapps 目录下。

六。配置 Tomcat

设置 JVM 参数

VIVO RDF 数据库的小部分复制到内存中，以便为快速的 Web 请求（在内存中的副本以及底层数据库保持同步进行编辑）。

VIVO 可能需要更多的内存比分配到 Tomcat 默认。大多数安装的 Tomcat，“setenv.sh”或“setenv.bat 的 Tomcat 的 bin 目录中的文件是一个方便的地方设置内存参数。如果这个文件不存在 Tomcat 的 bin 目录下，你可以创建它。例如：

export CATALINA_OPTS="-Xms512m -Xmx512m - XX:MaxPermSize=128m"

设置 Tomcat 的，分配初始堆 512 兆，512 兆字节，最大堆和 PermGen 空间 128 兆。较低的值可能就足够了，特别是对于小试装置。

如果 VIVO 执行过程中遇到的一个 OutOfMemoryError，它可以纠正，增加堆参数和重新启动 Tomcat。

设置安全限制

VIVO 是一个多线程的 Web 应用程序，可能需要比你的 Linux 安装的默认配置下，允许多个线程。确保安装可以支持以下编辑到/ etc /security/ limits.conf 中所需的线程数：

apache hard nproc 400

tomcat6 hard nproc 1500

设置 URI 编码

为了 VIVO Tomcat 上正确处理国际字符，您必须配置 Tomcat 接受编码 UTF-8，以符合标准的 URI。的编辑 Tomcat 的 conf/ server.xml 中添加以下属性到每个连接器元素：

URIEncoding="UTF-8". <Server ...> <Service ...>

<Connector ... URIEncoding="UTF-8"/> ...
</Connector> </Service> </Server>

创建上下文时要小心

每个在 VIVO 的分布（VIVO 和 Solr）的 webapps 包括“context.xml”的文件中，包含了一些信息，web 应用的部署。

允许你覆盖这些内容片段，通过添加上下文元素“server.xml 中”。如果你决定这样做，那就确保您的新的 Context 元素包括必要的部署参数重写的上下文片段。

七。启动 Tomcat

Tomcat 的安装可以启动运行 startup.sh 或 Tomcat 的 bin 目录下的 startup.bat。浏览器指向 http://localhost:8080/vivo，来测试应用程序。

在启动时，VIVO 会运行一些诊断测试。如果检测到的一个问题是正常的 VIVO 页面将重定向到启动状态页面描述的问题。您可以停止 Tomcat，试图解决这个问题，并着手从步骤五“启动状态”页面。

如果 Tomcat 没有启动，或对 VIVO 应用程序是打不开的，检查 Tomcatlogs 目录中的文件。错误消息通常在 TOMCAT /logs/ catalina.out 中或 tomcat/logs/ vivo.all.log 的或 tomcat/logs/ localhost.log 的中发现。

八。登录并添加 RDF 数据

如果启动成功，你会看到一个可喜的消息告诉你，你已经成功安装了 VIVO。单击“Login”接右上角附近，登录与你 rootUser.emailAddress 成立第四步。根帐户的初始密码是的“rootPassword”（不带引号）。在第一次登录时，系统会提示您选择一个新的密码，并验证它第二次。登录完成后，被选中的搜索索引，如果它是空的，OA 构建完整的索引将在后台触发，以确保整个网站的完整功能。

登录后，您将编辑选项菜单。在这里，你可以创建 OWL 类，对象的属性，数据属性，配置的数据显示。目前，任何你想使您的网站上可见的类必须是一类组的一部分，任何个人都必须有一个的 rdfs：label。有一些类和物理性能的知名度和显示选项。

在“Advanced Data Tools”，然后单击“Add/Remove RDF Data.”请注意，VIVO 外目前最好的 OWL-DL 的本体，只是有限的支持纯 RDF 数据。您可以输入一个 URL 指向你希望加载或从您的本地计算机上的文件上传到 RDF 数据。确保“add RDF""单选按钮被选中。你也可能会要勾选“create classgroups automatically.”。

单击“Index”选项卡页面右上角的导航栏会显示一个简单的索引的基础知识。

查看更多配置 VIVO 摄入数据，并手动添加在 http://vivoweb.org/support 数据文件。

九。设置联系人的电子邮件地址（如果使用“Contact Us”的形式）

如果你已经配置了应用程序使用“Contact Us”功能的步骤 IV（email.smtpHost），你也将需要添加一个电子邮件地址。VIVO 应用、这是电子邮件的接触形式提交。它可以是一个

服务器列表或个人的电子邮件地址。

作为一个系统管理员登录。导航“Site Admin”表的内容（链接，在右侧的头）。转到“Site Information”（在“Site Configuration”）。在“Site Information Editing Form，”输入电子邮件地址的功能在该领域“Contact Email Address”，并提交变更。如果设置 email.smtpHost 在第四步在这一步并没有提供一个电子邮件地址，您的用户将收到一个 java 接口错误。

十，设置 Apache Tomcat 的连接器

建议如 mod_jk 的一个 Tomcat 连接器可用于确保该网站地址不包含端口号（如 8080），和一个额外的参考 Tomcat 的上下文名称（例如/VIVO）。
这将使 VIVO 在“http://example.com”，而不是“http://example.com:8080/vivo”
使用 mod_jk 连接允许 Tomcat 和主 Web 服务器之间的通信。 Apache 的网站上的快速启动 HOWTO 介绍几种流行的 Web 服务器服务器的最低配置。
mod_jk 连接上述设置完成后，您将需要修改 Tomcat 的 server.xml 中（位于[tomcat 根]/ conf / 中），以应对来自 Apache 的请求通过连接器。看为的<Connector>指令并添加以下属性：
connectionTimeout="20000" maxThreads="320" keepAliveTimeout="20000"
Locate the <Host name="localhost"...>指令和更新内容如下：

```
<Host name="localhost"
    appBase="webapps"
    DeployOnStartup="false"
    unpackWARs="true" autoDeploy="false"
    xmlValidation="false"
    xmlNamespaceAware="false">
<Alias>example.com</Alias>
<Context
    path=""
    docBase="/usr/local/tomcat/webapps/vivo"
    reloadable="true"
    cookies="true" >
<Manager
    pathname="" />
<Environment
    type="java.lang.String"
    override="false"
    name="path.configuration"
    value="deploy.properties"
/>                          </Context>                                                    ...
```

十一。使用 VIVO 外部认证系统

VIVO 可以配置工作像 Shibboleth 的 CUWebAuth 的外部验证系统。

VIVO 必须只能通过一个 Apache HTTP 服务器。 Apache 服务器将被配置为调用外部认证系统。当用户完成身份验证，Apache 服务器将通过一个网络 ID，VIVO，来识别用户。

VIVO 有该用户的帐户，用户将被记录在该帐户的权限。在一个帐户的情况下，VIVO 将尝试找到与该用户相关联的一个页面。如果发现这样一个页时，用户可以登录到编辑自己的个人资料信息。

配置 Apache 服务器

您的机构将提供您与设置外部认证制度的说明。 Apache 服务器必须进行配置，以确保页面在 VIVO。当用户到达这个安全页面，Apache 服务器将调用外部认证系统。

安全页面 VIVO，这被命名为：/ loginExternalAuthReturn

当您的指示要求的位置，你应该使用这个值。

配置 VIVO

要启用外部身份验证，VIVO 需要三个值在 deploy.properties 文件。

•HTTP 标头的名称，将持有的外部用户的网络 ID。

当用户完成验证过程，Apache 服务器的 HTTP 请求的标头成一个把用户的网络 ID。从你的机构的指令应该告诉你这头被用于此目的。

你需要告诉 VIVO，HTTP 标头的名称。插入这样一行在 deploy.properties 文件：

externalAuth.netIdHeaderName = [the header name]

例如：

externalAuth.netIdHeaderName = remote_userID

•“Login”按钮的文本。

开始验证过程中，用户会点击一个按钮，在 VIVO 登录表单。你需要告诉 VIVO 该按钮应该出现在什么样的文字。

将这样一行在 deploy.properties 文件：externalAuth.buttonText=[文字]例如：您的登录按钮

externalAuth.buttonText = Log in using BearCat Shibboleth

VIVO 的登录表单将显示一个按钮标有 "Log in using BearCat Shibboleth".

•关联用户个人资料页。

VIVO 会联想到的用户个人资料页面，因此用户可以编辑自己的个人资料数据。 VIVO 将搜索的数据模型匹配的属性，用户的网络 ID（属性的值必须是一个字符串或无类型文本）一个人。你需要告诉 VIVO 什么属性应该用于匹配。插入这样一行在 deploy.properties 文件：

selfEditing.idMatchingProperty = [the URI of the property]

例如：

selfEditing.idMatchingProperty = http://vivo.mydomain.edu/ns#networkId