Leibniz-Zentrum für Psychologische Information und Dokumentation (ZPID)

Projektdokumentation

Datawiz - Ein Assistenzsystem für das Management psychologischer Forschungsdaten

Ronny Bölter

13. März 2018

Inhaltsverzeichnis

1 Einleitung **3**

2 Software **4**
2.1 Spring Tool Suite(Eclipse) . . . 4
2.2 Versionierung und kollaboratives Arbeiten . . . 4
2.2.1 Git . . . 4
2.2.2 Github.com . . . 4
2.3 Apache Tomcat/MySQL/PHP . . . 4
2.4 Minio Cloud Storage . . . 4
2.5 Verwendete Java Libraries . . . 5
2.5.1 Spring MVC . . . 5
2.5.2 Spring Security . . . 5
2.5.3 log4j . . . 5
2.5.4 JUnit . . . 5
2.5.5 MySQL Connectors (mysql-connector-java) . . . 5
2.5.6 Hibernate Validator (hibernate-validator) . . . 5
2.5.7 Java Bean Validation (javax-validation) . . . 5
2.5.8 JavaMail (javax-mail) . . . 5
2.5.9 GSON (google-gson) . . . 6
2.5.10 iText 7 . . . 6
2.5.11 openCSV . . . 6
2.5.12 Minio Java SDK . . . 6
2.5.13 dom4j . . . 6
2.5.14 Apache ODF Toolkit . . . 6
2.6 Verwendete Javascript Bibliotheken . . . 6
2.6.1 jQuery . . . 6
2.6.2 dropzone.js . . . 6
2.6.3 DataTables . . . 6

3 Einrichtung: Lokale Entwicklungsumgebung **7**
3.1 Java SE Development Kit . . . 7
3.1.1 Linux . . . 7
3.1.2 Windows 7-10 . . . 7
3.2 Spring Tool Suite . . . 9
3.3 Github DataWiz Repositorium . . . 9
3.4 DataWiz Import (Maven) . . . 10
3.5 SPSS-Java-IO Bibliothek . . . 11
3.6 MySQL Datenbank . . . 13
3.7 Minio (Multi-Cloud Object Storage) . . . 13
3.8 DataWiz Einstellungen (datawiz.properties) . . . 13
3.9 Lokaler Tomcat 8 Server . . . 15
3.10 Starten der Anwendung . . . 16

4 Einrichtung: Live System **17**
4.1 Ubuntu Server Installation . . . 17
4.2 Minio (Multi-Cloud Object Storage) . . . 17
4.2.1 Minio User . . . 17
4.2.2 TLS Verbindung . . . 17
4.2.3 Minio Installation . . . 18
4.2.4 Auto und Restart per Systemd . . . 19
4.3 MySQL Datenbank . . . 19
4.4 Tomcat Server . . . 19
4.5 Datawiz Deployment . . . 19

5 DataWiz: Programmierrichtlinien **20**
5.1 Logging 20
5.2 Formatting 20
5.3 JavaDoc 20

6 DataWiz: Implementierung **20**

Linkverzeichnis **21**

1 Einleitung

Die Implementierung von DataWiz erfolgt in Java (Enterprise Edition) unter Hinzunahme von weiteren, momentan gängigen Technologien. Hierzu zählt im Frontend Bereich (View) neben den üblichen Standards wie HTML5, CSS3 und JavaScript vor allem die Hinzunahme von Bootstrap, jQuery und einigen kleineren jQuery Bibliotheken, welche im späteren Verlauf erklärt werden. Bei der Entwicklung des Backend Bereiches (Model & Controller) wurde neben Java/Java EE vor allem weite Teile des Spring MVC Frameworks verwendet. Zusätzlich kommt die Spring Security Erweiterung zum Einsatz um das System sicher vor Angriffen und möglichem Diebstahl von elektronisch gespeicherten Daten zu machen. Detailtiere Erklärungen zu den genutzten Frameworks und Libraries sind im Kapitel 2 zu finden.

2 Software

2.1 Spring Tool Suite(Eclipse)

„The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications. It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, and comes on top of the latest Eclipse releases.
Included with the Spring Tool Suite is the developer edition of Pivotal tc Server, the drop-in replacement for Apache Tomcat that's optimized for Spring. With its Spring Insight console, tc Server Developer Edition provides a graphical real-time view of application performance metrics that lets developers identify and diagnose problems from their desktops.
The Spring Tool suite supports application targeting to local, virtual and cloud-based servers. It is freely available for development and internal business operations use with no time limits, fully open-source and licensed under the terms of the Eclipse Public License.“[19]

2.2 Versionierung und kollaboratives Arbeiten

Softwareentwicklung setzt meistens voraus, dass Teile der Anwendung kollaborativ entwickelt werden und vor allem im Falle eines Fehler durch Versionskontrolle auf einen funktionierenden Stand zurückgesetzt werden kann. Deswegen ist es wichtig sich im Vorfeld mit der Einrichtung eines solchen Systems zu befassen. Da das ZPID über einen GitHub Account verfügt wurde entschlossen diese Plattform zu nutzen. GitHub hat neben den Versionierungs- und Kollaborationsfeatures den Vorteil, dass man die entwickelte Software ohne großen Mehraufwand der Öffentlichkeit zur Verfügung stellen kann und diese durch eine möglicherweise entstehende Community weiter entwickelt und verbessert wird. Zur Nutzung von GitHub wird die Software Git benötigt.

2.2.1 Git

„Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.“[6]
Git ist bereits als Plug-In in der Spring Tool Suite enthalten und muss nicht installiert werden. Die Einrichtung von Git innerhalb der STS wird im Abschnitt 3 erklärt.

2.2.2 Github.com

GitHub[9] ist ein webbasierter Online-Dienst, der auf Git basiert und als Repositorium für Software-Projekte dient. Der Quelltext von DataWiz wird auf www.github.com in einem eigenen Repositorium[21] zur Ansicht, aber auch zur Weiterentwicklung bereitgestellt. Für den korrekten Umgang mit GitHub steht eine ausführliche Hilfe[10] bereit.

2.3 Apache Tomcat/MySQL/PHP

Für den Betrieb von DataWiz wird neben einem Apache Tomcat Server[5], auch ein Datenbank Server (MySQL) und optional ein Apache Server mit PHP (für phpMyAdmin) benötigt.

2.4 Minio Cloud Storage

Das Minio[1][11] System besteht aus 2 Komponenten. Einem Datei Server welcher Dateien verwaltet. Im Falle von DataWiz werden alle hoch-geladenen Dateien in diesem System abgelegt. Dies betrifft die Zusatzmaterialien und Datenmatrizen(CSV/SPSS). Im Prinzip ist der Server auch mit Public-Cloud Anbietern austauschbar, wie beispielsweise Amazon S3 oder Google Cloud. Hierfür müssen nur die Konfigurationseinstellung für die Minio-Anbindung in der datawiz.properties Datei angepasst werden.

[1] An open source object storage server with Amazon S3 compatible API, , written in Go and open sourced under Apache License Version 2.0. Build cloud-native applications portable across all major public and private clouds.

2.5 Verwendete Java Libraries

In dieser Unterkategorie werden die verwendeten Java Bibliotheken aufgelistet. Der Import dieser Bibliotheken und die Informationen über die aktuell genutzten Versionen befinden sich in der pom.xml im Hauptverzeichnis der DataWiz-Anwendung. Ebenso befinden Sich dort die Links zu den Maven-Repositorien der einzelnen Bibliotheken.

2.5.1 Spring MVC

„The Spring Web MVC framework provides Model-View-Controller (MVC) architecture and ready components that can be used to develop flexible and loosely coupled web applications. The MVC pattern results in separating the different aspects of the application (input logic, business logic, and UI logic), while providing a loose coupling between these elements.“

2.5.2 Spring Security

„Spring Security is a powerful and highly customizable authentication and access-control framework. It is the de-facto standard for securing Spring-based applications.“

2.5.3 log4j

„log4j ist ein Framework zum Loggen von Anwendungsmeldungen in Java. Innerhalb vieler Open-Source- und kommerzieller Softwareprodukte hat es sich über die Jahre zu einem De-facto-Standard entwickelt.“

2.5.4 JUnit

„JUnit ist ein Framework zum Testen von Java-Programmen, das besonders für automatisierte Unit-Tests einzelner Units (Klassen oder Methoden) geeignet ist.“

2.5.5 MySQL Connectors (mysql-connector-java)

„MySQL offers standard database driver connectivity for using MySQL with applications and tools that are compatible with industry standards ODBC and JDBC. Any system that works with ODBC or JDBC can use MySQL.“

2.5.6 Hibernate Validator (hibernate-validator)

„Hibernate Validator allows to express and validate application constraints. The default metadata source are annotations, with the ability to override and extend through the use of XML. It is not tied to a specific application tier or programming model and is available for both server and client application programming.“

2.5.7 Java Bean Validation (javax-validation)

„JSR 380 is a specification of the Java API for bean validation, part of JavaEE and JavaSE, which ensures that the properties of a bean meet specific criteria, using annotations such as @NotNull, @Min, and @Max.
This version requires Java 8 or higher, and takes advantage of new features added in Java 8 such as type annotations, and supports new types like Optional and LocalDate.“

2.5.8 JavaMail (javax-mail)

„JavaMail ist eine Java-Programmierschnittstelle zum Plattform- und Protokoll-unabhängigen Senden und Empfangen von E-Mails. JavaMail unterstützt dabei die Standards SMTP, POP3 und IMAP. Die JavaMail API ist Teil der Java-EE-Plattform, kann aber auch als optionales Package von der Java Standard Edition aus verwendet werden. Seit 2. März 2009 ist JavaMail OpenSource und kann als JavaMail API Referenz-Implementierung über das Projekt Kenai bezogen werden“

2.5.9 GSON (google-gson)

„Gson is a Java library that can be used to convert Java Objects into their JSON representation. It can also be used to convert a JSON string to an equivalent Java object. Gson can work with arbitrary Java objects including pre-existing objects that you do not have source-code of. There are a few open-source projects that can convert Java objects to JSON. However, most of them require that you place Java annotations in your classes; something that you can not do if you do not have access to the source-code. Most also do not fully support the use of Java Generics. Gson considers both of these as very important design goals.“

2.5.10 iText 7

„iText ist eine freie Programmbibliothek zur Erzeugung und Bearbeitung von PDF-Dateien mittels der Programmiersprache Java. Sie wurde von Bruno Lowagie, Paulo Soares und anderen entwickelt. Die Software wird unter der GNU Affero General Public License (AGPL) vertrieben.“

2.5.11 openCSV

„opencsv is an easy-to-use CSV (comma-separated values) parser library for Java. It was developed because all the CSV parsers at the time didn't have commercial-friendly licenses. Java 7 is currently the minimum supported version.“

2.5.12 Minio Java SDK

„The Minio Java Client SDK provides simple APIs to access any Amazon S3 compatible object storage server.“

2.5.13 dom4j

„dom4j ist eine in der Programmiersprache Java geschriebene Open-Source-Programmierschnittstelle (API) für den Zugriff und die Verarbeitung von XML-Dokumenten.“

2.5.14 Apache ODF Toolkit

„The Apache ODF Toolkit is a set of Java modules that allow programmatic creation, scanning and manipulation of Open Document Format (ISO/IEC 26300 == ODF) documents. Unlike other approaches which rely on runtime manipulation of heavy-weight editors via an automation interface, the ODF Toolkit is lightweight and ideal for server use.“

2.6 Verwendete Javascript Bibliotheken

2.6.1 jQuery

„jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.“[13]

2.6.2 dropzone.js

„DropzoneJS is an open source library that provides drag'n'drop file uploads with image previews.“[20]

2.6.3 DataTables

„DataTables is a plug-in for the jQuery Javascript library. It is a highly flexible tool, based upon the foundations of progressive enhancement, and will add advanced interaction controls to any HTML table.“[14]

3 Einrichtung: Lokale Entwicklungsumgebung

In diesem Kapitel wird erklärt wie eine Entwicklungsumgebung für DataWiz installiert wird und DataWiz importiert und lauffähig gemacht wird. Da DataWiz eine Java (Enterprise Edition) Anwendung ist, besteht die Möglichkeit die (Weiter-)Entwicklung unter Windows, Linux und MacOS vorzunehmen. Als IDE wird die Spring Tool Suite(STS) empfohlen und deren Installation auch im Unterkapitel 3.2 genauestens erklärt. Es ist aber auch möglich mit anderen Umgebungen (Netbeans, Eclipse oder IntelliJ IDEA) zu arbeiten, aber dafür werden keine Anleitungen bereitgestellt, ebenso wie für das Betriebssystem MacOS. Ja nach Betriebssystem unterscheidet sich die Einrichtungsaufwand etwas.

3.1 Java SE Development Kit

Zur Entwicklung von DataWiz wird das JDK in der Version 8 verwendet, da DataWiz für das Bauen der .war Dateien Maven benutzt und die tools.jar des JDK von diversen Bibliotheken benötigt wird. Die Installation des JDK, bzw. dessen korrekte Verwendung ist unter Windows etwas anders als unter Linux/Mac OS.

3.1.1 Linux

Die Installation des JDK ist unter Linux (hier am Beispiel mit apt unter Ubuntu 16.04) mit einem Befehl erledigt:

```
sudo apt-get install openjdk-8-jdk
```

3.1.2 Windows 7-10

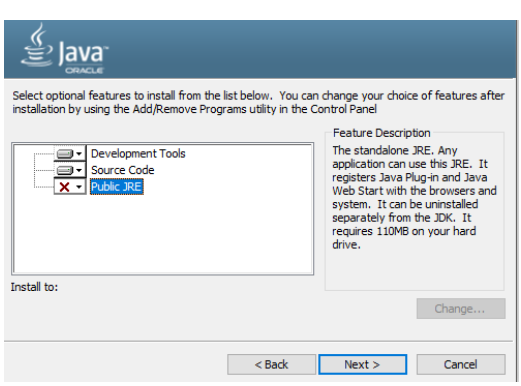
Das Verhalten von Java Installationen und deren Nutzung kann unter Windows etwas unübersichtlich sein, da die Installation von verschiedenen JDKs und JREs unter Windows bei der normalen Installation immer neue Ordner anlegt und die alten Versionen beibehält. Dies kann im schlimmsten Falle dazu führen, dann man irgendwann etliche Versionen von Java auf seinem Gerät vorfindet. Damit es beim Import von DataWiz nicht zu Java Problemen kommt wie beispielsweise fehlender .jar Dateien (tools.jar) ist es notwendig Eclipse mit einer JDK zu starten. Hierfür gibt es mehrere Möglichkeiten:

- **Startparameter setzen:**
  Um Eclipse/STS mit einem speziellen JDK zu starten muss in der eclipse.ini (STS.ini bei der Spring Tool Suite) folgender Eintrag am Anfang der Datei eingefügt werden, indem der Pfad zum JDK definiert ist, mit dem Eclipse gestartet werden soll.

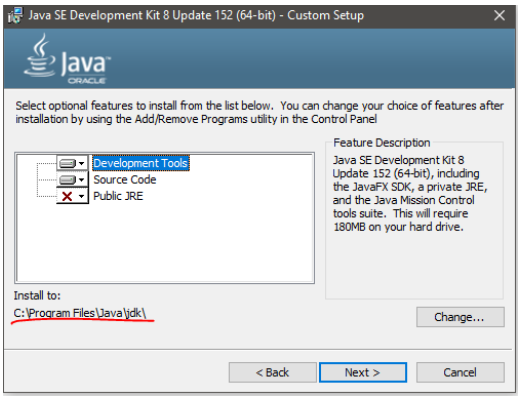
  ```
  -vm
  <Path to JDK Folder>\bin\javaw.exe
  ```

  Dieses Vorgehen hat den Nachteil, das eventuell irgendwann mit einem veralteten JDK gearbeitet wird, oder beim Löschen des JDK der Pfad in der .ini Datei geändert werden muss.

- **Umgebungsvariablen setzen**
  Java Versionen sind in der Regel abwärtskompatibel, daher ist es durchaus sinnvoll nur ein JDK/JRE auf der Maschine zu betreiben. Hierfür werden erst einmal alle auf dem Gerät befindlichen Versionen gelöscht und das neuste JDK heruntergeladen[15] und installiert. Da die Umgebungsvariablen manuell gesetzt werden, ist es nicht notwendig eine öffentliche (public) JRE zu installieren und deswegen sollte diese während des Installationsprozesses abgewählt werden (Abbildung 1a). Zur besseren Übersicht wird empfohlen den Pfad zum JDK anzupassen(Abbildung 1b), da dieser immer mit der Versionsnummer versehen ist. Bei einem Update des JDK muss die neue Version nur wieder in diesen Ordner installiert und die gesetzten Pfade der Umgebungsvariablen müssen nicht erneut angepasst werden.



(a) Abwahl der Public JRE

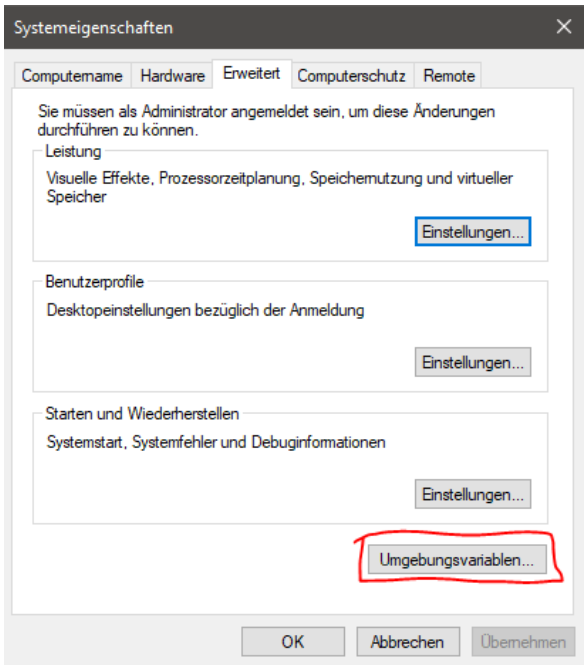


(b) Speicherpfad des JDK ändern

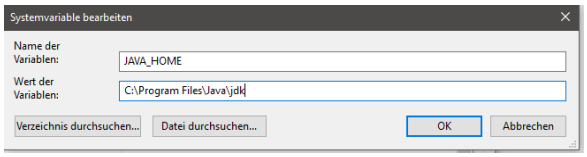
Abbildung 1: JDK Installation

Um die Umgebungsvariablen zu Erstellen/Bearbeiten:

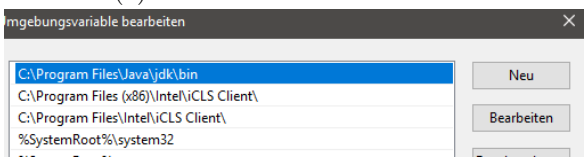
- Tastenkombination [Win -R], um den Dialog Ausführen zu öffnen.
- *SystemPropertiesAdvanced.exe* eingeben und mit *Enter* bestätigen.
- Im neu geöffneten Fenster(Abbildung 2a) auf Umgebungsvariablen klicken
- Unter „Systemvariablen“ die Variable JAVA_HOME entweder anlegen (Neu...), oder falls vorhanden bearbeiten (Bearbeiten...) und den Pfad zum installierten JDK Ordner unter „Wert der Variable“ eintragen. Siehe Abbildung 2b.
- Nach dem gleichen Prinzip die Variable „Pfad“ anlegen, oder bearbeiten.(Abbildung 2c)



(a) Systemeigenschaften



(b) Variable JAVA_HOME setzen



(c) Variable Path setzen

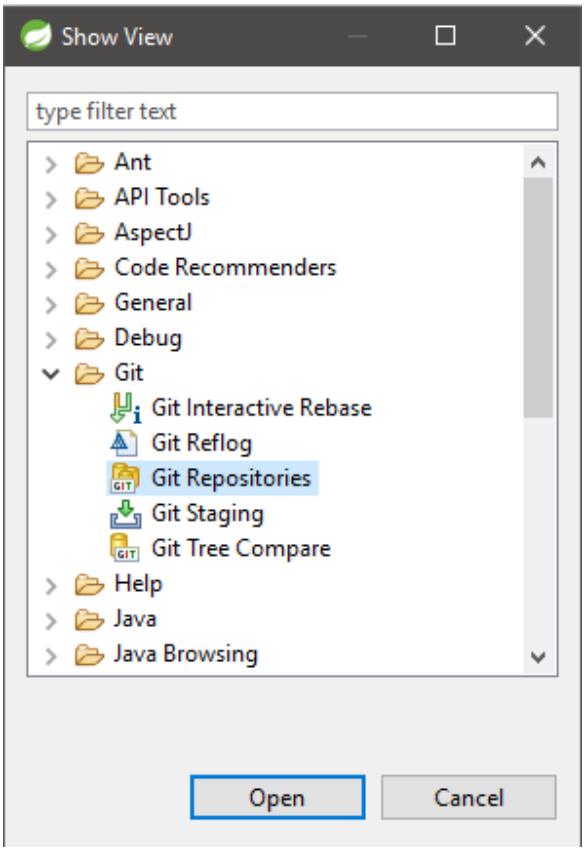
Abbildung 2: Setzen der Java Umgebungsvariablen

3.2 Spring Tool Suite

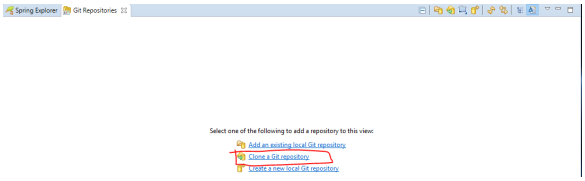
Die Spring Tool Suite (STS) von der Webseite[19] herunterladen, in ein beliebiges Verzeichnis entpacken und danach kann die STS über die STS.exe (Windows) oder STS.sh (Linux) im sts-x.x.x.RELEASE Ordner gestartet werden. Bei Bedarf kann auch eine normale Instanz von Eclipse genutzt werden, wichtig ist aber, dass die Java-EE, Maven und Git Erweiterungen installiert sind, damit DataWiz importiert werden kann.

3.3 Github DataWiz Repositorium

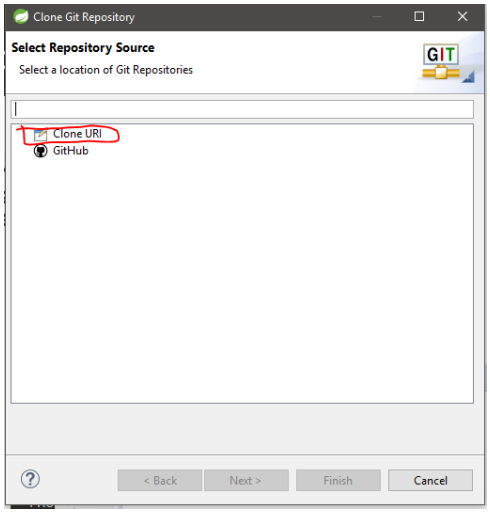
Um das DataWiz Github Repositorium hinzuzufügen muss in der STS vorerst die Repositorien Ansicht (Abbildung 3b) hinzugefügt werden. Hierzu muss im Hauptmenü unter dem Reiter *Window* → *Show View* → *Other...* ausgewählt und dann in dem *Show View* Fenster (Abbildung 3a) unter Git die *Git Repositories* View ausgewählt und mit *Open* bestätigt werden. Im *Git Repositories* Tab (Abbildung 3b) ,welches normalerweise unten in der rechten Spalte geöffnet wird, *Clone a Git repository* auswählen. Im nächsten Fenster(Abbildung 3c) gibt es nun die Möglichkeiten eine URL zu klonen (*Clone URI*) oder in GitHub nach Repositorien zu suchen (*Github*). Die Suche funktioniert nur mit öffentlichen Repositorien und nicht mit privaten. Für DataWiz sollte *Clone URI* genutzt werden, da die URL[21] zum Repositorium bekannt ist. Im nächsten Fenster(Abbildung 3d) muss nun die URL zum DataWiz bei URI eingefügt werden. Die Authentifizierung ist notwendig wenn das Repositorium privat ist und man als Contributor eingetragen ist. Mit *Next* zum nächsten Fenster, indem man den Branch auswählen kann in dem man arbeiten möchte. DataWiz nutzt momentan nur den Master, deswegen ist dieser vorausgewählt. Mit *Next* zum nächsten Fenster, um die lokalen Speichereinstellungen gegeben falls anzupassen, und mit *Finish* den Importvorgang finalisieren.



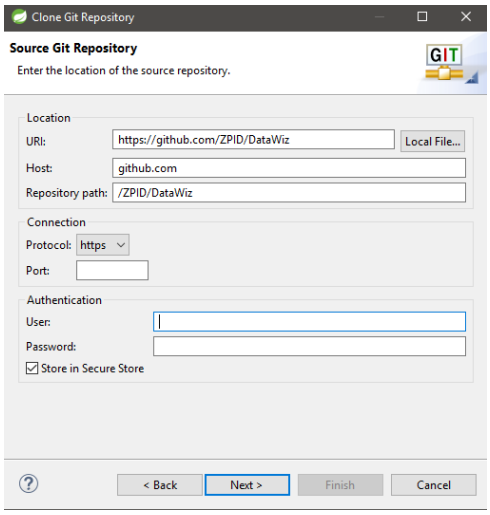
(a) Auswahl der Git Repositorium Ansicht



(b) Git Repositorium Ansicht in der STS



(c) Git Repositorium per URL klonen

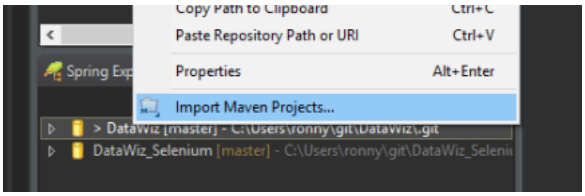


(d) GitHub Einstellungen

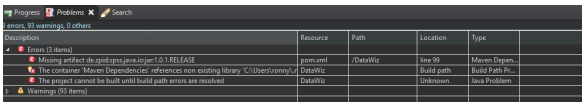
Abbildung 3: DataWiz Github Repositorium einbinden und klonen

3.4 DataWiz Import (Maven)

Nach dem Hinzufügen des Github Repositoriums (Kapitel 3.3), ist dieses nun in der Repositorien Ansicht verfügbar(Abbildung 4a) und DataWiz kann nun als Maven Projekt importiert werden. Hierzu einfach ein Rechtsklick auf das DataWiz Repositorium und im folgenden Dialog die „/DataWiz/pom.xml“ Datei auswählen und mit *Finish* den Importprozess starten. Maven wird nun alle benötigten Bibliotheken aus den öffentlichen Archiven herunterladen und danach das Projekt validieren. Dies dauert für gewöhnlich einen Augenblick. Am Ende der Validierung sollte es im optimalen Fall nur drei Fehlermeldungen geben wie in Abbildung 4b. Die SPSS-JAVA-IO Bibliothek muss manuell nachinstalliert werden, dieser Schritt wird im Kapitel 3.5 behandelt.



(a) Maven Import starten



(b) Java Error aufgrund der fehlenden SPSS-JAVA-IO Bibliothek

Abbildung 4: Maven Import von DataWiz

3.5 SPSS-Java-IO Bibliothek

Die SPSS IO Funktionalität wurde in einem eigenständigen Projekt entwickelt und wird DataWiz per .jar Datei zur Verfügung gestellt. Das Problem ist, dass DataWiz Maven zur Bibliotheksverwaltung nutzt, die SPSS-JAVA-IO aber nicht in den öffentlichen Repositorien (bspw. Maven Central) gelistet ist. Deswegen muss die Bibliothek manuell zu DataWiz hinzugefügt werden.

Hierzu wird in der STS das Projekt mit Maven mit dem Ziel „install:install-file“ ausgeführt. Im Vorfeld müssen die benötigten Dateien von GitHub heruntergeladen[4] und das 7z (7zip) Archiv entpackt werden. Danach in der STS auf das Projekt DataWiz mit einem Rechtsklick das Kontextmenü öffnen und dort unter *Run As* → *Maven build...* das „Edit Configuration“(Abbildung 5) Fenster öffnen um die Import Parameter zu definieren.

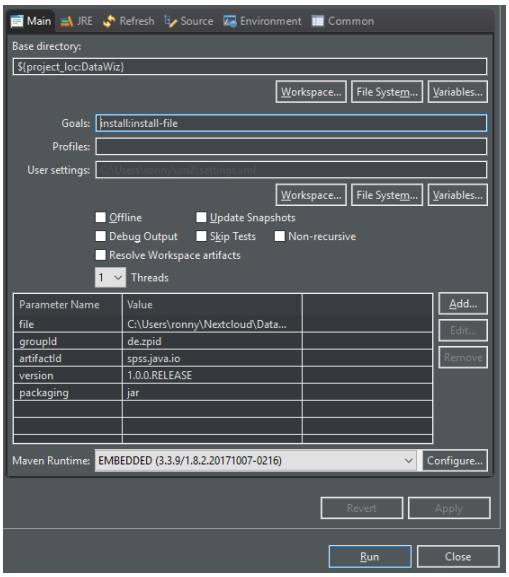


Abbildung 5: Maven build Einstellungen

Dazu gibt es die folgenden 2 Möglichkeiten:

1. Manuelle Parametereingabe (Beispiel Abbildung 5):

- Goals: install:install-file
- Add Parameter: Name: *file*; Value: *Vollständiger Pfad zur .jar Datei*
- Add Parameter: Name: *groupId*; Value: *de.zpid*
- Add Parameter: Name: *artifactId*; Value: *spss.java.io*
- Add Parameter: Name: *version*; Value: *1.0.0.RELEASE* (Kann sich logischerweise ändern)
- Add Parameter: Name: *packaging*; Value: *jar*

2. Parameter aus .pom Datei:

- Goals: install:install-file
- Add Parameter: Name: *file*; Value: *Vollständiger Pfad zur .jar Datei*
- Add Parameter: Name: *pomFile*; Value: *Vollständiger Pfad zur .pom Datei*

Sind alle Parameter gesetzt, wird der Build mit *Run* gestartet und sollte eine ähnliche Ausgabe wie im Listing 1 erzeugen. Falls „Error“ Meldungen erscheinen, bitte die Parameter überprüfen. Dazu müssen diese nicht neu eingeben werden, sondern können über: „*Rechtsklick auf das Projekt DataWiz* → *Run As* → *Run Configurations* → *Konfiguration auswählen*“ erneut geladen werden.

```
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building DataWiz 1.0.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-install-plugin:2.4:install-file (default-cli) @ DataWiz ---
[INFO] Installing C:\Users\ronny\Nextcloud\DataWiz\SPSS\SPSS\3.0\SPSS-3.0.jar to
    C:\Users\ronny\.m2\repository\de\zpid\spss.java.io\1.0.0.RELEASE\spss.java.io-1.0.0.RELEASE.jar
[INFO] Installing C:\Users\ronny\AppData\Local\Temp\mvninstall9171890829130493852.pom to
    C:\Users\ronny\.m2\repository\de\zpid\spss.java.io\1.0.0.RELEASE\spss.java.io-1.0.0.RELEASE.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 0.719 s
[INFO] Finished at: 2018-01-11T15:11:28+01:00
[INFO] Final Memory: 16M/491M
[INFO] ------------------------------------------------------------------------
```

Listing 1: Konsolenausgabe des Maven Kommandos „install:install-file“

Wichtig: Bei einer Änderung der Versionsnummer (Name des heruntergeladenen 7z Ordners) werden die Dateien von Maven in einem anderen Unterordner angelegt. Das wird auch durch über Konsolenausgabe angezeigt(Listing 2):

```
...
[INFO] Installing C:\Users\ronny\Nextcloud\DataWiz\SPSS\SPSS\3.0\SPSS-3.0.jar to
    C:\Users\ronny\.m2\repository\de\zpid\spss.java.io\1.0.0.RELEASE\spss.java.io-1.0.0.RELEASE.jar
[INFO] Installing C:\Users\ronny\AppData\Local\Temp\mvninstall9171890829130493852.pom to
    C:\Users\ronny\.m2\repository\de\zpid\spss.java.io\1.0.0.RELEASE\spss.java.io-1.0.0.RELEASE.pom
...
```

Listing 2: Anzeige der aktuellen Version

Diese Versionsnummer muss mit der Version in der pom.xml des Projektes übereinstimmen. Dazu wird diese geöffnet *(STS:Package Explorer → DataWiz → pom.xml)* und die property „<spss.java.io.version>“ gesucht und der Inhalt mit aktuell genutzten Version ausgetauscht (Listing 3).

```
...
<properties>
   <spss.java.io.version>x.x.x.RELEASE</spss.java.io.version>
   ...
```

Listing 3: Anpassung der SPSS-Java-IO Version in der DataWiz pom.xml

Nun sollten alle Java Error Meldungen im Projekt behoben sein, falls dies nicht der Fall ist, sollte ein „rebuild“ von DataWiz erfolgen. Dieses Prozess kann initialisiert werden durch: *Project (STS Hauptmenü) → Clean → DataWiz auswählen → Clean).*

Wichtig:
Die SPSS-Java-IO Bibliothek ist ein Java-Wrapper für die von IBM bereitgestellten „SPSS Statistics Input/Output Module“, da diese in C geschrieben sind. Dies bedeutet, dass die native Bibliotheken für Windows und Linux heruntergeladen werden müssen und der Speicherort der SPSS-Java-IO Bibliothek bekanntgegeben werden muss. Während der Entwicklung der Java Bibliothek wurden die Input/Output Module der SPSS Version 24 verwendet und diese sind auch ausgiebig getestet worden. Deswegen wird empfohlen diese Version zu nutzen[7]. SPSS Version 25[8] ist gegen Ende der Entwicklung veröffentlicht worden, es gab zwar keine Fehler bei der Umstellung und auch die Input/Output Funktionen funktionieren, aber eine intensive Testphase ist bis jetzt nicht durchgeführt worden. Nach dem Download der Version 24[7] oder 25[8], müssen diese entpackt werden und entweder der Ordner in dem sie sind umbenannt, oder ein neuer erstellt werden mit dem Namen „native“. In diesem liegen die nativen Bibliotheken in Unterordnern wie in Abbildung 6. Momentan werden von der SPSS-Java-IO Bibliothek Linux 64Bit und Windows 32/64Bit unterstützt, deswegen werden nur die Ordner win32, win64 und lin64 benötigt. Der Quelltext der SPSS-Java-IO Bibliothek wird auf GitHub[3] bereitgestellt.Nachdem der „native“ Ordner vorhanden ist und die Bibliotheken eingefügt sind, muss der Speicherort des Ordners noch in den DataWiz Einstellungen eingetragen werden(Abschnitt 3.8).

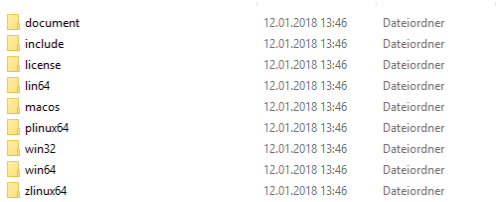


Abbildung 6: Inhalt des native Ordners

3.6 MySQL Datenbank

Die Installation einer MySQL-Datenbank für Linux wird im Unterkapitel 4.3 beschrieben. Für Windows gibt es 2 Möglichkeiten:

1. **Windows 10: Linux-Subsystem:(empfohlen)**
   Windows 10 bietet die Möglichkeit ein schlankes Linux System als Subsystem zu starten. Die Installation erfolgt in 2 Schritten:

   - Die Powershell als Administrator öffnen und folgenden Code ausführen:

     ```
     Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
     ```

     Starten Sie den Computer neu, wenn Sie dazu aufgefordert werden.
   - Den Windows Store öffnen und das gewünschte Subsystem installieren. Momentan ist erhältlich: Ubuntu, openSUSE Leap 42 und SUSE Linux Enterprise Server 12.

   Nachdem das Subsystem installiert wurde, kann es über das App-Icon im Startmenü gestartet und die Installation einer MySQL Datenbank wie im Unterkapitel 4.3 durchgeführt werden.

2. **WAMP** Falls das Betriebssystem kein Subsystem unterstützt(Windows 8 oder 7), dann kann ein WAMP-Server (Apache, MySQL, PHP on Windows) installiert werden. Hierzu der Installationsanleitung auf der Webseite[1] folgen. Nachdem der WAMP-Server installiert wurde kann man per phpMyAdmin eine neue Datenbank anlegen und die DataWiz-Tabellen importieren. Diese sind im DataWiz GitHub Repositorium[21] im Ordner „database“ zu finden. Die Zugangsdaten zur erzeugten Datenbank (am besten mit einem Nutzer ohne root-Rechte) sind in den Datawiz Einstellungen(Unterkapitel 3.8) zu hinterlegen.

3.7 Minio (Multi-Cloud Object Storage)

Falls ein eigener Minio-Server als Dateiablage betrieben werden soll, kann die Anleitung aus Kapitel 4.2 verwendet werden. Die Installation des Live-Systems und des Testsystems sind in diesem Falle identisch. Es kann sogar ein Minio für beide Systeme genutzt werden. Die Trennung der abgelegten Dateien durch verschiedenen Systeme kann über den Prefix(Kapitel 3.8) geregelt werden. Für die Anbindung an einen Public-Cloud Dienstleister (AWS, Google Cloud), müssen in den DataWiz Einstellungen (Kapitel 3.8) unter „Minio SETTINGS“ die Zugangsdaten von dem Anbieter verwendet werden.

3.8 DataWiz Einstellungen (datawiz.properties)

DataWiz und auch die Anbindungen zu den externen Diensten, wie beispielsweise Minio oder die MySQL Datenbank, lassen sich in der Datei „datawiz.properties“ konfigurieren. Nachdem das Projekt in der STS im Projektexplorer einzusehen ist, befindet sich im Ordner */src/main/resources* die Datei „datawiz_example.properties“. Diese muss umbenannt (*Rechtsklick auf die Datei → Refactor → Rename*) in „datawiz.properties“. Danach die Datei im Editor öffnen.

Die Datei ist in verschiedene Bereiche gegliedert:

- **APPLICATION SETTINGS:** (Allgemeine Einstellungen)

  - **application.url**: Hier ist anzugeben unter der DataWiz betrieben werden soll. Das funktioniert auch mit IP-Adressen, falls keine URL zur Verfügung steht. Falls dieser Eintrag fehlerhaft ist, oder fehlt, können beim Export von PDF Dateien Fehler auftreten.
  - **organisation.admin.email**: E-Mail Adresse, die dem Nutzer in Falle von Fehlern als Kontaktadresse angezeigt wird.
  - **session.timeout**: Regelt die Laufzeit der Session, also die Zeit die ein Nutzer eingeloggt bleibt wenn er keine Aktionen auf der Seite durchführt.
  - **application.name**: Beinhaltet den Namen der Anwendungen, dieser wird z.B. für den Versand von E-Mails genutzt.
  - **organisation.name**: Name der Organisation welche die DataWiz Instanz betreibt
  - Alle weiteren Einstellungen dienen nur grafischen Elementen und ermöglichen den Austausch der Logos.

- **DATABASE SETTINGS:**
  - **dataSource.driverClassName**: Als Datenbank für DataWiz wurde MySQL gewählt, deswegen ist hier der MySQL-Treiber(com.mysql.cj.jdbc.Driver) voreingestellt. Es ist möglich andere relationale Datenbanksysteme zu verwenden.
  - **dataSource.url**: URL zur Datenbank. Hier sind ein paar Parameter voreingestellt die je nach Datenbanksetting und Verbindung notwendig sein könnten.
  - **dataSource.username** & **dataSource.password**: Datenbanknutzer Zugangsdaten

- **SMTP SETTINGS:**(E-Mail Versand)
  - **mail.smtp.host**: URL zum Mail Server
  - **mail.smtp.starttls.enable**: An(*true*)/Aus(*false*) der Transport Layer Security (TLS) Kommunikationsverschlüsselung
  - **mail.smtp.auth**: Erwartet der Mail-Server eine Authentifizierung zum Versand, dann *true*, ansonsten *false*
  - **mail.smtp.port**: SMTP Port
  - **mail.smtp.username** & **mail.smtp.password**: Zugangsdaten für den Mail-Server
  - **mail.set.from**:Name des Absenders
  - **mail.set.content**: Wie der Content der Mail dargestellt werden soll. HTML voreingestellt.

- **Minio SETTINGS:**(Storage Backend)
  - **minio.url**: URL zum Minio Server, oder falls die Speicherung bei einem externen Anbieter erfolgt, dann hier die URL zum Cloud-Storage(z.B. Amazon AWS oder Google Cloud) .
  - **minio.access.key**: Access-Key wird entweder vom Anbieter vergeben, oder beim Betrieb einen eigenen Minio-Servers während der Installation erzeugt(siehe Kapitel 4.2).
  - **minio.secret.key**: Secret-Key wird entweder vom Anbieter vergeben, oder beim Betrieb einen eigenen Minio-Servers während der Installation erzeugt(siehe Kapitel 4.2).
  - **minio.cert.selfsigned**: *true* wenn ein selbst erstelltes Zertifikat zur Übertragungsverschlüsselung(TLS) zwischen Minio-Server und DataWiz genutzt wird.
  - **minio.bucket.prefix**: Prefix der Ordner im Cloud-Storage. Hat den Vorteil, dass mehrere Anwendungen ein Daten-Backend nutzten können.

- **SPSS Native library path:**(Pfad zu den nativen SPSS-Bibliotheken)
  Die hier verwendeten Pfade müssen auf den Ordner zeigen, indem der „native“ Ordner liegt, der bei der Installation angelegt wurde. (Kapitel 3.5).
  - **spss.absoluth.path.windows**: Pfad zum native Ordner auf Windows Systemen

  - **spss.absoluth.path.unix**: Pfad um native Ordner auf Unix Systemen

- **TrashMail-Blacklist API:**(API zum Abfangen von Trash-Mail Adressen)
  DataWiz verwendet eine externe freie API[2] zum Erkennen von Trash-Mail Adressen, um zu verhindern, dass sich Spammer und Bots auf DataWiz anmelden können.
  - **trashmail.blacklist.api**: URL(https://v2.trashmail-blacklist.org/check/json/) zur API ist eingetragen.

- **FOLDER SETTINGS API:**
  DataWiz benötigt Ordner zum Ablegen von temporären Dateien und Log-Dateien.
  - **folder.root.unix**: Pfad zum DataWiz Hauptverzeichnis in Linux-Systemen
  - **folder.root.windows**: Pfad zum DataWiz Hauptverzeichnis in Windows-Systemen
  - **folder.temp.dir**: Name des Ordners mit den temporären Dateien (liegt im DataWiz Hauptverzeichnis).

- **BETA SETTINGS:**
  Diese Einstellungen versetzten DataWiz in den „Beta-Status“. Dies bedeutet, das bestimmte Texte angezeigt werden, die vor einem Datenverlust waren, da es sich um eine nicht stabile Version handelt.
  - **datawiz.beta**: *true*, um den Beta-Status zu aktivieren
  - **application.logo.url.beta**: Pfad zu einem bestimmten Logo, welches nur im Beta-Status angezeigt wird.

3.9 Lokaler Tomcat 8 Server

Für das Deployment von Java-EE Anwendung wird ein Applikations-Server benötigt. Für DataWiz wird der Tomcat 8 Server empfohlen. Es gibt auch die Möglichkeiten GlassFish, Jetty, Pivotal tc Server Developer Edition v3.2 oder andere Server zu verwenden, aber während der Entwicklung von DataWiz kam Tomcat 8 zum Einsatz und ist deswegen auf diesem ohne großen Konfigurationsaufwand lauffähig. Inzwischen liegt die Tomcat Version 9 vor, aber mit dieser war ein Starten des aktuellen DataWiz Build nicht möglich. Eventuell wird hier im Laufe der Weiterentwicklung nachgebessert.
Um den Tomcat in die Entwicklungsumgebung(STS) einzubinden, muss dieser von der Apache Webseite[5] heruntergeladen werden. Danach die gepackte Datei in einen beliebigen Ordner entpacken und die STS öffnen. In der STS gibt es unten links einen Reiter „Server“ (Abbildung 7). Falls dieser Reiter nicht vorhanden ist, dann diesen im Hauptmenü unter *Window* → *Show View* → *Servers* aktivieren.

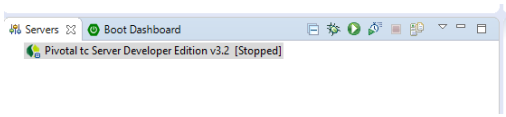


Abbildung 7: Serveransicht in der STS

Um den Tomcat zu importieren, *Rechtsklick das Servers Fenster* → *New* → *Server* und den folgenden Dialog wie folgt durchführen:

- **Select the Server Type:** Apache → Tomcat v8.5 Server
- **Servers host name:** localhost (voreingestellt)
- **Server Name:** Anzeigename des Servers im Server Reiter, von daher kann er beliebig sein, oder einfach den voreingestellten Verwenden.
- **Next**
- **Name:** Übernimmt den Server Name aus dem vorangegangenem Fenster
- **Tomcat installation directory:** Pfad zum entpackten Tomcat Ordner

- **JRE:** Workbench default JRE
- **Next**
- DataWiz Anwendung von **Available** per *Add* nach **Configured** übertragen und per *Finish* den Dialog beenden.

Nun sollte der neue Tomcat-Server im Servers-Reiter zur Verfügung stehen.

3.10 Starten der Anwendung

Um die Anwendung zu starten muss nun nur noch der Tomcat Server gestartet werden. Hierzu im Servers-Reiter den Tomcat auswählen entweder per *Start the Server* Icon, oder *CTRL-ALT-R*, oder *Rechtsklick* → *Start* den Server starten. Nun startet in der STS Konsole die Log-Ausgabe. Wichtig hierbei ist, das neben dem Tomcat-Log-Einträgen auch DataWiz Eintrage fehlerfrei und ohne Error Nachrichten durchlaufen. Der Server ist vollständig gestartet wenn am Ende die Meldung **INFORMATION: Server startup in ... ms** erscheint. Nun kann DataWiz im beliebigen Web-Browser über die Adresse: http://localhost:8080/DataWiz/ aufgerufen werden.

```
...
2018-01-17 14:18:21,661 [localhost-startStop-1] INFO: datawiz.configuration.DataWizConfiguration ---
    viewResolver succesfully loaded
2018-01-17 14:18:21,675 [localhost-startStop-1] INFO: datawiz.configuration.DataWizConfiguration ---
    localeResolver succesfully loaded
2018-01-17 14:18:21,696 [localhost-startStop-1] INFO: datawiz.configuration.DataWizConfiguration ---
    multipartResolver succesfully loaded
2018-01-17 14:18:21,767 [localhost-startStop-1] INFO: web.context.ContextLoader --- Root WebApplicationContext:
    initialization completed in 5093 ms
Jan 17, 2018 2:18:21 PM org.apache.catalina.core.ApplicationContext log
INFORMATION: Initializing Spring FrameworkServlet 'dispatcher'
2018-01-17 14:18:21,773 [localhost-startStop-1] INFO: web.servlet.DispatcherServlet --- FrameworkServlet
    'dispatcher': initialization started
2018-01-17 14:18:21,776 [localhost-startStop-1] INFO: context.support.AnnotationConfigWebApplicationContext ---
    Refreshing WebApplicationContext for namespace 'dispatcher-servlet': startup date [Wed Jan 17 14:18:21 CET
    2018]; parent: Root WebApplicationContext
2018-01-17 14:18:21,851 [localhost-startStop-1] INFO: web.servlet.DispatcherServlet --- FrameworkServlet
    'dispatcher': initialization completed in 78 ms
Jan 17, 2018 2:18:21 PM org.apache.coyote.AbstractProtocol start
INFORMATION: Starting ProtocolHandler ["http-nio-8080"]
Jan 17, 2018 2:18:21 PM org.apache.coyote.AbstractProtocol start
INFORMATION: Starting ProtocolHandler [äjp-nio-8009"]
Jan 17, 2018 2:18:21 PM org.apache.catalina.startup.Catalina start
INFORMATION: Server startup in 18015 ms
```

Listing 4: Anzeige der aktuellen Version

4 Einrichtung: Live System

4.1 Ubuntu Server Installation

TODO

4.2 Minio (Multi-Cloud Object Storage)

In diesen Abschnitt geht es um die Installation eines Minio Servers zum Speichern sämtlicher Projekt- und Studienmaterien ebenso wie den Datenmatrizen der hochgeladenen Datensätze. Es ist möglich das Minio-System auf dem Server zu installieren, auf dem auch DataWiz betrieben wird, oder das Minio-System als standalone Server zu betreiben und mit DataWiz über eine sichere TLS-Verbindung zu koppeln. Die zweite Variante hat den Vorteil, dass Minio von Webserver abgekoppelt in einem sicheren Netzwerkbereich betrieben werden kann und damit eventuell vor Angriffen besser geschützt ist. Vor allem, da der Betrieb von Minio nur einen offenen Port benötigt und somit entsprechende Firewall-Regeln genutzt werden können.
In den folgenden Abschnitten wird die Installation anhand einer standalone Lösung mit einem eigenen Unix Benutzer für das Minio-System (ohne Root-Rechte) erklärt. Falls das Minio-System auf dem selben Server installiert werden soll, auf dem auch die DataWiz-Anwendung (Tomcat-Server) betrieben wird, dann können sie direkt in Abschnitt 4.2.3 einsteigen.

Bitte beachten Sie, dass alle Schritte in den Abschnitten 4.2.2, 4.2.3 und 4.2.4 mit dem aus Abschnitt 4.2.1 erzeugten User und in dessen Home-Verzeichnis durchgeführt werden. Wird ein andere Nutzer für die Installation verwendet, so müssen alle „/home/minio/“ Einträge in den Anweisung entsprechend abgeändert werden in „/home/<username>/“!

4.2.1 Minio User

Aus Sicherheitsgründen sollte es vermieden Minio mit einem User zu starten, der Root-Rechte anfordern kann. Deswegen wird in diesem Abschnitt ein neuer Nutzer mit Home-Ordner angelegt. Dazu folgendes ausführen:

```
adduser minio
```

Neues Passwort vergeben und im nächsten Schritt bestätigen:

```
Geben Sie ein neues UNIX-Passwort ein:
```

Optionale Informationen vergeben:

```
Changing the user information for minio
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
```

Erstellung des Nutzers abschließen, indem man folgendes mit „j“ bestätigt:

```
Sind diese Informationen korrekt? [J/n]
```

4.2.2 TLS Verbindung

Dieser Schritt ist nur notwendig, wenn das Minio-System vom Webserver abgekoppelt wird und auf einem separaten Server läuft. Es wird empfohlen die Verbindung zwischen dem Webserver und Minio-Server zu verschlüsseln. Dazu bietet Minio eine TLS-Übertragungsverschlüsselung an. Die folgenden Befehle sind für eine „self-signed“ Zertifikat Konfiguration. Es ist aber auch möglich „trusted“ Zertifikate zu verwenden, oder im Falle, dass der zukünftige Minio Server eine eigene URL hat, auch mit Let's Encrypt Zertifikaten zu arbeiten. Weitere Anleitungen befinden sich in der Minio.io Dokumentation[12] unter Minio Server/How to secure access to Minio server with TLS.

Das Erstellen eines Zertifikates benötigt einen Nutzer mit Root-Rechten, aber da Minio als Ablageort der zu verwendeten Zertifikate immer vom Home Ordner des Nutzers ausgeht mit dem der Service gestartet wird, müssen die zu erstellenden Zertifikate dort abgelegt werden.

Wechsel in das Home-Verzeichnes des Nutzers mit dem Minio ausgeführt wird:

```
cd /home/minio/
```

Erstellen eines Unterordners mit dem Namen „.minio“:

```
sudo mkdir .minio
```

Wechsel in den erstellten .minio Unterordner:

```
cd .minio/
```

Erstellen eines Unterordners mit dem Namen „certs“:

```
sudo mkdir certs
```

Wechsel in den erstellten certs Unterordner:

```
cd certs
```

Erstellen eines privaten Schlüssels mit openssl:

```
sudo openssl genrsa -out private.key 2048
```

Erstellen eines öffentlichen Zertifikates mit openssl:
„/C=US/ST=state/L=location/O=organization/CN=domain“ sollten angepasst werden, im Falle des ZPID sieht das wie folgt aus: „/C=DE/ST=rlp/L=germany/O=zpid/CN=zpid.de“

```
sudo openssl req -new -x509 -days 3650 -key private.key -out public.crt -subj
    "/C=US/ST=state/L=location/O=organization/CN=domain"
```

Dem Nutzer Minio die Rechte an dem Ordner .minio samt Unterordner und enthaltenen Dateien vergeben:

```
sudo chown -R minio:minio /home/minio/.minio
```

Nachdem das Zertifikat erstellt wurde, ist es notwendig dieses in die Zertifikatsliste des Server zu importieren auf dem DataWiz läuft, da ansonsten der Handshake fehlschlägt. **Deswegen werden die folgende Befehle nicht auf dem Minio-Server ausgeführt, sondern auf dem DataWiz Webserver.**

public.crt auf den Webserver vom Minio-Server laden:

```
scp user@host:/home/minio/.minio/certs/public.crt minio_server.crt
```

Zertifikat in den ca-certificates kopieren:

```
sudo cp minio_server.crt /usr/local/share/ca-certificates/
```

Zertifikatsliste des Servers updaten:

```
sudo update-ca-certificates
```

```
minio.cert.selfsigned=true
```

4.2.3 Minio Installation

Login mit user „minio“ und folgendes ausführen.

Verzeichnis „minio-server“ in „/home/minio/“ anlegen:

```
mkdir minio-server
```

In das Verzeichnis "minio-server"wechseln:

```
cd minio-server
```

Aktuellen Minio Server herunterladen:

```
wget https://dl.minio.io/server/minio/release/linux-amd64/minio
```

Minio Server ausführbar machen:

```
chmod +x minio
```

Minio Server mit dem Verzeichnis /home/minio/minio-data/ßtarten (Server legt Verzeichnis an):

```
./minio server /home/minio/minio-data
```

Es sollte folgender Output kommen. Dabei ist es wichtig den **AccessKey und den SecretKey zu notieren**, da diese später für die Anbindung von DataWiz und auch für die Weboberfläche von Minio benötigt werden.

```
...
AccessKey: FZ2FDGDFGHHAAAFDFSDF
SecretKey: eU9UlFZsdf7s8dfs+sfdsdf-sfs!sdf
...
```

4.2.4 Auto und Restart per Systemd

Damit der Minio-Server beim System Neustart mit startet und vor allem bei einen Crash automatisch neustartet muss ein Service Eintrag im Systemd eingetragen werden.
Erstellen der minio.service Datei:

```
sudo nano /etc/systemd/system/minio.service
```

Diese nun mit folgenden Inhalt füllen:

```
[Unit]
Description=Minio Server
After=network.target

[Service]
ExecStartPost=/bin/echo "Minio started"
Type=simple
ExecStart=/home/minio/minio-server/./minio server /home/minio/minio-data
User=minio
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Neuen Service einschalten per:

```
sudo systemctl enable minio.service
```

Server rebooten:

```
sudo reboot now
```

4.3 MySQL Datenbank

4.4 Tomcat Server

4.5 Datawiz Deployment

5 DataWiz: Programmierrichtlinien

5.1 Logging

5.2 Formatting

5.3 JavaDoc

6 DataWiz: Implementierung

Linkverzeichnis

[1] Romain Bourdon. WampServer. http://www.wampserver.com/en/. (Eingesehen am 08.01.2018).

[2] Tobias Dillig. Trashmail-Blacklist. https://trashmail-blacklist.org/. (Eingesehen am 08.01.2018).

[3] Ronny Bölter (Leibniz Institute for Psychology Information). SPSS-Java-IO. https://github.com/ZPID/SPSS_utils. (Eingesehen am 05.01.2018).

[4] Ronny Bölter (Leibniz Institute for Psychology Information). SPSS-Java-IO Download. https://github.com/ZPID/SPSS_utils/tree/master/mvn%20install. (Eingesehen am 08.01.2018).

[5] Apache Software Foundation. Apache Tomcat®. https://tomcat.apache.org/download-80.cgi. (Eingesehen am 08.01.2018).

[6] Junio C. Hamano, Shawn O. Pearce, Linus Torvalds, and many more. git –local-branching-on-the-cheap. https://git-scm.com/. (Eingesehen am 10.11.2015).

[7] IBM. SPSS-Utils Download version 24. https://www-01.ibm.com/marketing/iwm/iwm/web/reg/download.do?source=swg-tspssp&lang=en_US&S_PKG=ds&cp=UTF-8. (Eingesehen am 08.01.2018).

[8] IBM. SPSS-Utils Download version 25. https://dal05.objectstorage.softlayer.net/v1/AUTH_14da1a30-d001-4a2c-9060-cadaf68ff44d/yp-ngp-spss-dal05/SPSSTools/Statistics25/IO_Module_for_SPSS_Statistics_25.zip. (Eingesehen am 08.01.2018).

[9] GitHub Inc. GitHub. https://github.com/. (Eingesehen am 10.11.2015).

[10] GitHub Inc. GitHub Help. https://help.github.com/. (Eingesehen am 05.01.2018).

[11] Minio Inc. Minio. https://minio.io/. (Eingesehen am 05.01.2018).

[12] Minio Inc. Minio Documentation. https://docs.minio.io/. (Eingesehen am 15.12.2017).

[13] The jQuery Foundation. jQuery. https://jquery.com/. (Eingesehen am 10.11.2015).

[14] SpryMedia Ltd. DataTables. https://datatables.net/. (Eingesehen am 08.01.2018).

[15] Oracle. Java SE Development Kit 8 Downloads. http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html. (Eingesehen am 08.01.2018).

[16] Mark Otto and Jacob Thornton. Bootstrap. http://getbootstrap.com/. (Eingesehen am 10.11.2015).

[17] Pivotal Software. Spring Framework. http://projects.spring.io/spring-framework/. (Eingesehen am 10.11.2015).

[18] Pivotal Software. Spring Security. http://projects.spring.io/spring-security/. (Eingesehen am 10.11.2015).

[19] Pivotal Software. Spring Tool Suite™. https://spring.io/tools/sts. (Eingesehen am 10.11.2015).

[20] Matias Meno www.colorglare.com. DROPZONE^{JS}. http://www.dropzonejs.com/. (Eingesehen am 08.01.2018).

[21] ZPID. DataWiz GitHub Repositorium. https://github.com/ZPID/DataWiz/. (Eingesehen am 05.01.2018).