The James Hutton Institute

The James Hutton Institute
Information & Computational Sciences

Germinate 3

Documentation

Version 3.4.0

September, 2017

Developers:

Paul Shaw
Sebastian Raubach
Iain Milne
Gordon Stephen
David Marshall

Contents

1 Introduction 1

2 Setup 2
2.1 Requirements . . . 2
2.2 Database setup . . . 2
2.3 Building Germinate from source . . . 2
2.3.1 Additional requirements . . . 2
2.3.2 Downloading the source code . . . 2
2.3.3 Configuration of source code . . . 3
2.3.4 Building the source code . . . 3
2.4 Using the Germinate installation script . . . 4
2.5 Deploying an existing Germinate .war file . . . 4
2.6 Running the Germinate Docker image . . . 4
2.7 Germinate Gatekeeper Admin . . . 5
2.8 Logging . . . 5

3 Configuration 6
3.1 Germinate Menu . . . 10
3.1.1 Structure . . . 10

4 Features 13
4.1 Internationalization and Localization . . . 13
4.2 News . . . 13
4.3 Available pages . . . 13
4.3.1 About Germinate . . . 13
4.3.2 About Project . . . 13
4.3.3 Accessions Overview . . . 14
4.3.4 Accessions for Collecting Site . . . 14
4.3.5 Acknowledgements . . . 14
4.3.6 Administrator Configuration . . . 14
4.3.7 Allele Frequency Datasets . . . 15
4.3.8 Allele Frequency Export . . . 15
4.3.9 Allele Frequency Results . . . 15
4.3.10 Climate Data . . . 15
4.3.11 Climate Datasets . . . 16
4.3.12 Chemical Compounds . . . 16
4.3.13 Chemical Compound Details . . . 16
4.3.14 Chemical Compound Datasets . . . 16
4.3.15 Chemical Compound Data . . . 17
4.3.16 Cookie . . . 17
4.3.17 Data Statistics . . . 17
4.3.18 Dataset Overview . . . 17
4.3.19 Experiment Details . . . 18
4.3.20 Genotype Datasets . . . 18
4.3.21 Genotype Export . . . 18
4.3.22 Geographic Search . . . 18
4.3.23 Group Preview . . . 19
4.3.24 Groups . . . 19

4.3.25 Image Gallery 19
4.3.26 Locations 19
4.3.27 Home 20
4.3.28 Institutions 20
4.3.29 Maps 20
4.3.30 Marked Items 20
4.3.31 Marker Details 21
4.3.32 Mega Environments 21
4.3.33 News 21
4.4 User registration 21
4.5 Dataset licenses 22
4.6 Security 22

5 Structure 24
5.1 GitHub project 24
5.1.1 Project structure 25
5.1.1.1 Config properties 25
5.1.1.2 Germinate xml 25
5.1.1.3 Web xml 25
5.1.2 Web structure 26
5.2 Database 26

6 Customization 27
6.1 Customizing the Germinate theme 27
6.2 Internationalizing your pages 27
6.2.1 Parametrization and Plural Forms 28
6.2.2 Notes 28
6.3 Adding a new language 28
6.3.1 Notes 29
6.4 Internationalized files and images 29
6.4.1 Passing parameters via the URL 30
6.4.2 Custom Bootstrap Theme 30

7 Data Import 31
7.1 Germinate Daim 31
7.2 Germinate Data Templates 32

8 Statistics 33

9 Troubleshooting 34
9.1 Germinate links in download files are not HTTPS 34

10 Funding 35

11 Technologies 36
11.1 JavaScript libraries 36
11.2 Web frameworks 37
11.3 Font frameworks 37
11.4 Java libraries 37

1 Introduction

Germinate is a generic platform for the storage and dissemination of multiple data types associated with genetic resource collections. Examples of the types of data that Germinate currently supports are passport, phenotypic, field trial, pedigree, genetic and geographic location data. Our aim is to keep Germinate as flexible as possible and we will add support for additional data types over time.

Germinate not only acts as storage for experimental data but offers a user-friendly interface into the data and acts as a backend for analysis tools such as Helium for pedigree visualization, our graphical genotyping application Flapjack [1] and CurlyWhirly for the simple display of xyz coordinate data such as PCO and PCA. All these tools are available from `https://ics.hutton.ac.uk`. We have also prioritised the development of tools to allow users to export data in a variety of formats for analysis in external applications such as R.

Germinate 3 builds on the existing Germinate 2 platform an adds additional tools, functionality and introduces a much more simple installation process over its predecessor. These changes also allow us to deploy Germinate 3 both within server and desktop environments which was difficult with Germinate 2. We have also removed the limitation of requiring Linux, Germinate 3 is now compatible with any environments that have Apache Tomcat. Germinate takes the Germinate platform away from its Perl roots and has been completely rewritten using the GWT Web Toolkit (GWT) from Google. The underlying database is as it was and while we recommend using MySQL should be compatible with a number of relational database management systems with minor changes.

The move from Perl to Java and GWT has allowed us to introduce a number of new features. Examples include full internationalization and localization support which allows us to offer Germinate in multiple languages if suitable translations exit as well as offering a more advanced and responsive web-interface and user access control.

We hope that you find Germinate useful and our vision is that Germinate forms platform on to which additional tools can be added over time. The common platform means that any additional functionality can be rolled out to all other Germinate installations which makes it both a flexible and continually evolving platform to help meet the needs of the genetic resources community.

If you encounter any problems, have ideas for features that would be useful to include in Germinate or just want to chat about the system then we would love to hear from you and we can be contacted in a number of ways. By email on `germinate@hutton.ac.uk`, or you can write to us at:

Germinate,
Information & Computational Sciences,
The James Hutton Institute,
Invergowrie,
Dundee,
DD2 5DA, UK.

We also have a website (`https://ics.hutton.ac.uk/get-germinate`) that we keep up to date with current developments of Germinate and all our other analysis and visualization tools and you can follow us on Twitter `@cropgeeks`.

2 Setup

In this section, we will explain how to get Germinate up and running. There are several different ways to achieve this. Depending on your expertise and requirements, you can choose the option that suits you best. Building from source requires basic knowledge of the command line.

2.1 Requirements

We try to keep the requirements of Germinate as basic as possible. In order to run Germinate you will need to have the following applications available on your server:

- *Apache Tomcat* (7.0.28 or above) to run the web application
- *Java* (8 or above) to run Apache Tomcat and the Germinate server side code
- *MySQL* (or *MariaDB*) (5.6.1 or above) database to hold the data

Please make sure that you have the required applications installed before continuing.

2.2 Database setup

Germinate and Germinate Gatekeeper have their own separate databases. Each one needs to be set up before being able to use Germinate. We provide a simple SQL script that takes an empty database and sets up all the tables and views that Germinate requires. This script is included in the ”database” subfolder in the source code of Germinate and Germinate Gatekeeper (cf. Section 2.3.2). Simply run this script against the empty Germinate and Germinate Gatekeeper databases respectively.

2.3 Building Germinate from source

Building Germinate from source is the most flexible way of getting Germinate up and running. It allows you to make use of all of Germinate's customization options and tailor it specifically to your needs.

2.3.1 Additional requirements

In addition to the basic requirements, you will need Apache Ant (1.9.1 or above) to be installed on the system you are building Germinate on.

2.3.2 Downloading the source code

We host the Germinate and Germinate Gatekeeper code on GitHub since version 3.4.0. They are available here:

`https://github.com/germinateplatform/`

To download the code, please use your favourite Git client and create a clone of the latest release of Germinate and Germinate Gatekeeper respectively.

2.3.3 Configuration of source code

Before you start configuring your instance of Germinate, take a copy of the folder `instance-stuff/template` into the folder `instance-stuff/<your germinate instance>`. This folder and its contents are explained in Section 5.1.

Both applications contain a file called `build.xml` which is the Apache Ant build script as well as an associated `build.properties` file. These files are used to compile and deploy the application. Edit the `build.properties` file and replace the placeholder username and password with your Apache Tomcat username and password. Replace the placeholder for your web server as well.

```
# Ant properties for building the GWT app
project.name=germinate-templace
project.root=jhi.germinate.Germinate

instance.files=./instance-stuff/<your germinate instance>

tomcat.manager.url=http://<Your web server>:8080/manager/text
tomcat.manager.username=<Username>
tomcat.manager.password=<Password>
```

The next file that needs editing is the `config.properties` file. In the case of Gatekeeper this file is located in the root directory. In the case of Germinate, you can find this file in the `instance-stuff/<your germinate instance>` folder.

Configure Gatekeeper in the following way:

```
database.server=<server holding germinate>
database.name=<database name>
database.useport=<is a port necessary?>
database.port=<port number>
database.username=<username, e.g. germinate3>
database.password=<password>
[...]
```

Finally, configure Germinate in the following way:

```
Germinate.Database.Server=<server holding germinate>
Germinate.Database.Name=<database name>
Germinate.Database.UsePort=<is a port necessary?>
Germinate.Database.Port=<port number>
Germinate.Database.Username=<username, e.g. germinate3>
Germinate.Database.Password=<password>

Gatekeeper.URL=<base url of gatekeeper>
Gatekeeper.Database.Name=<name of gatekeeper database>
Gatekeeper.Database.Server=<server holding gatekeeper>
Gatekeeper.Database.UsePort=<is a port necessary?>
Gatekeeper.Database.Port=<port number>
[...]
```

2.3.4 Building the source code

To build the application, simply run `ant` from the command line in the root directory of Germinate and Germinate Gatekeeper. Apache Ant will then run the build script and compile the source code and finally deploy it to your Apache Tomcat installation.

Make sure that the machine you're compiling the source on can communicate with the web server via HTTP (required to deploy the application).

After the build finishes successfully, you should be able to view Germinate in your browser. The address is based on your configuration. It should have this structure: `http://<Your web server>:8080/<project.name>`.

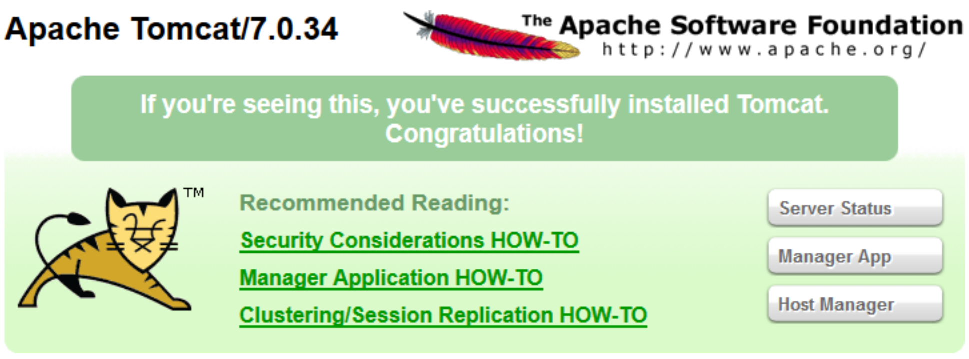

Apache Tomcat/7.0.34
The Apache Software Foundation
http://www.apache.org/
If you're seeing this, you've successfully installed Tomcat. Congratulations!
Recommended Reading:
Security Considerations HOW-TO
Manager Application HOW-TO
Clustering/Session Replication HOW-TO
Server Status
Manager App
Host Manager

Figure 1: Tomcat welcome screen

2.4 Using the Germinate installation script

Section 2.3 walked you through the process of building Germinate from source. If you would like to automate a low of the steps involved in this process, you can try and run the Germinate installation script. This script runs on Linux and is meant to be used when you are working with Germinate for the first time. It will take care of all the steps from downloading the source, setting up the database to building and deploying the source code.

Detailed information about the script are available on our website at:

```
https://ics.hutton.ac.uk/germinate/download-germinate/
```

2.5 Deploying an existing Germinate .war file

In case you have been provided with compiled versions of Germinate and Germinate Gatekeeper in the form of a `.war` file each as well as their database counterparts in your MySQL installation, then all you need to do is deploy them to Tomcat.
To deploy these applications, navigate to the following URL:

```
http://<Your web server>:8080
```

You should see something similar to Figure 1. Click on the button labelled ”Manager App” and you will be prompted to enter your credentials. Use the username and password defined in the file `tomcat-users.xml`. The following page will show all the applications running on Tomcat right now. Below this table, there is a section called ”Deploy” with subsection ”WAR file to deploy”. Simply select each of the two WAR files by clicking on the ”Browse” button and then click on ”Deploy”. Germinate and Germinate Gatekeeper should now be listed in the applications table at the top of the page.

Clicking on the `Path` link in the first column of the overview table should redirect you to Germinate and Germinate Gatekeeper respectively.

2.6 Running the Germinate Docker image

If none of the previous options seem to be for you, we distribute our demonstration version of Germinate as a Docker image that you can run and try out yourself. This image is meant to be used as a first point of contact with Germinate. It allows you to run Germinate locally without having to install it. You can then explore the web interface and try out all the features that Germinate has to offer.

Detailed instructions for this Docker image are available online at:

```
https://ics.hutton.ac.uk/germinate/germinate-docker-image/
```

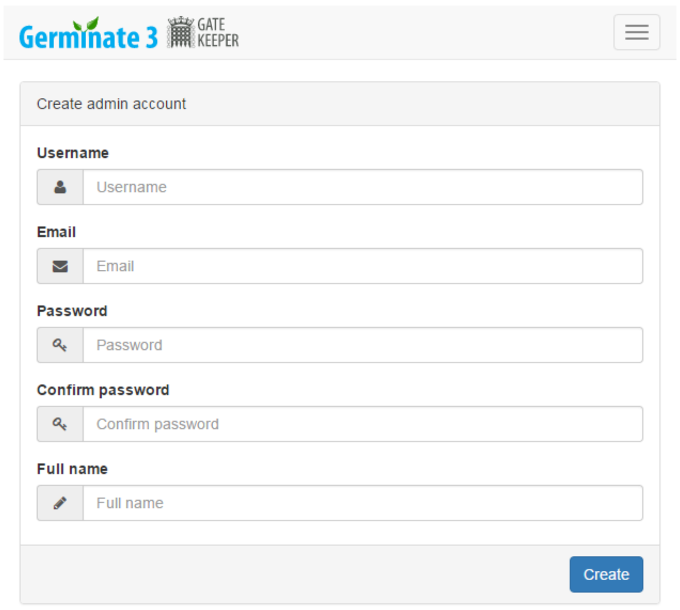


Figure 2: Create admin account page

2.7 Germinate Gatekeeper Admin

The first time you go to the Germinate Gatekeeper website, you'll see the form shown in Figure 2. Fill it in to create the initial admin account. After this, you will be able to log in to Gatekeeper and create other users, database systems and set their permissions.

2.8 Logging

Depending on your configuration (ref. `Germinate.Server.Logging.Enabled` in Section 3), Germinate will log all server-side exceptions. This can be useful when you need to debug your version of Germinate.

The log files are stored to this location: `<Tomcat Directory>/temp/logs/`. Each instance of Germinate uses its own log files to avoid mix-ups.

3 Configuration

This section will highlight some of the configurations of Germinate. The configurations are stored in the file `instance-stuff/<project.name>/config.properties`. Currently the following settings are available:

```
Germinate.Database.Username=<username for the database>
Germinate.Database.Password=<password for the database

Germinate.Database.Server=<server of the germinate database>
Germinate.Database.Name=<name of the germinate database>
Germinate.Database.Port=<port for database connection>

Gatekeeper.URL=<base url of gatekeeper>
Gatekeeper.Database.Server=<server of the gatekeeper database>
Gatekeeper.Database.Name=<name of the gatekeeper database>
Gatekeeper.Database.Port=<port for database connection>

Gatekeeper.BCrypt.Rounds=<number of rounds for the bcrypt hashing algorithm>
Gatekeeper.Registration.Enabled=<allow user registration?>
Gatekeeper.Registration.Needs.Approval=<user registration needs manual approval?>

Germinate.UseAuthentication=<users have to log in to see data?>
Germinate.CookieLifespanMinutes=<the lifespan of cookies in minutes>
Germinate.Debug=<enable to see sql queries run on the server>
Germinate.KeepTemporaryFileForHours=<how long should temporary files be kept?>
Germinate.UploadSizeLimitMB=<file size limit of uploads in MB>
Germinate.AvailablePages=<list of comma separated pages that are available for this instance of germinate>

Germinate.ShowHomeOnLogin=<show the content of the home page on the login page as well?>

Germinate.Server.Logging.Enabled=<should exceptions be logged on the server?>

Germinate.IsUnderMaintenance=<is the server under maintenance?>

Germinate.IsReadOnly=<is the database in read-only mode?>

Germinate.ExternalDataFolder=<optional external data directory outside tomcat>

Germinate.HideIdColumns=<should internal id columns be hidden from tables?>

Germinate.Gallery.Images.Per.Page=<determines the number of images per page>

GoogleAnalytics.Enabled=<should google analytics be enabled?>
GoogleAnalytics.TrackingId=<the google analytics tracking id>

CookieNotifier.Enabled=<should the cookie notifier for EU cookie law be enabled?>

Germinate.Template.CustomMenu=<the custom menu structure in XML format>

Germinate.Template.CategoricalColors=<colors used for some of the charts>
Germinate.Template.GradientColors=<colors used for the gradients>
Germinate.Template.Social.ShowFacebook=<show facebook share button?>
Germinate.Template.Social.ShowTwitter=<show twitter share button?>
Germinate.Template.Social.ShowGooglePlus=<show google+ share button?>
Germinate.Template.Logo.Contains.Link=<does the svg logo contain links?>
Germinate.Template.Show.Parallax.Banner=<show parallax image banner?>

Germinate.Template.EmailAddress=<contact email email address>

Germinate.Template.Title=<title of the germinate instance>
Germinate.Template.DatabaseName=<name of the germinate instance>
Germinate.Template.TwitterLink=<twitter link>

Path.Java=<path to the java installation>
Path.R=<path to the r installation>
```

All items starting with `Germinate.Database` have already been discussed in Section 2.3.3. Items starting with `Gatekeeper.Database` can be configured analogously.

We will now explain the meaning and possible settings of all the items. Non-optional properties are marked with "$\circ$" whereas "$*$" marks properties that are non-optional if you want to use the associated functionality. Unmarked properties either have a default value shown in **bold** or aren't crucial for Germinate to work properly.

`Germinate.Database.Username`$^\circ$ [String]
The MySQL username used for the Germinate database.

`Germinate.Database.Password`$^\circ$ [String]
The MySQL password associated with the username.

`Germinate.Database.Server`$^\circ$ [String]
The server MySQL is running on.

`Germinate.Database.Name`$^\circ$ [String]
The name of the Germinate database.

`Germinate.Database.Port`* [$x \in \{0, \ldots, 49151\}$]
The port number to use for the database connection.

`Gatekeeper.URL`* [URL]
The URL of Germinate Gatekeeper. This is required if the registration feature is enabled. Refer to property `Germinate.Registration.Enabled` and Section 4.4.

`Gatekeeper.Database.Server`* [String]
The server MySQL is running on.

`Gatekeeper.Database.Name`* [String]
The name of the Germinate Gatekeeper database.

`Gatekeeper.Database.Port`* [$x \in \{0, \ldots, 49151\}$]
The port number to use for the database connection.

`Gatekeeper.BCrypt.Rounds` [$x \in \{4, \ldots, \mathbf{10}, \ldots, 31\}$]Germinate Gatekeeper uses BCrypt to hash passwords. The number of rounds determines how long the hashing will take. A larger number will make it harder to use brute-force to find the password, but at the same time it will influence how long users will have to wait for their password to be verified. The default is a value of 10 which results in 2^{10} rounds. Increasing this to 11 will **double the runtime** to 2^{11}.

`Gatekeeper.Registration.Enabled` [true | **false**]Germinate and Gatekeeper support user registration. If this property is set to `true`, users will be able to register for access to your instance of Germinate. The registration will be accessible from the login page. See Section 4.4 for more details.

`Gatekeeper.Registration.Needs.Approval` [**true** | false]
If registration is enabled, there are two ways how users are approved for access to Germinate. If this property is set to `false`, users will automatically be approved and can start using Germinate right away. If this is not what you want, set this to `false` which will require you to approve requests manually through the Gatekeeper interface. Users will be notified with your decision via mail.

`Germinate.UseAuthentication` [true | **false**]
If set to `true`, the user will have to log in using the credentials stored in Germinate Gatekeeper, otherwise no login procedure is required.

`Germinate.CookieLifespanMinutes` [$x \in \mathbb{N}^+$; default: **1440**]
The lifespan of all cookies given in minutes.

`Germinate.Debug` [true | **false**]
Set to `true` if you want to see the SQL queries that are run against the database on each page. This is very handy when debugging or searching for errors. However, this should **never** be used in production use, since it exposes the database internals.

`Germinate.KeepTemporaryFileForHours` [$x \in \mathbb{N}^+$; default: **24**]
The amount of hours temporary files should be kept for. Temporary files are generated whenever the user wants to download parts of the database. They are kept locally at least for the given amount of hours and after this amount of time they will be deleted the next time a user requests any temporary file.

`Germinate.UploadSizeLimitMB` [Float; default **0.5**]
The file size limit of uploads in MB.

`GoogleAnalytics.Enabled` [true | **false**]Germinate supports Google Analytics. Page navigation as well as user interactions will be tracked if this property is set to `true`.

`GoogleAnalytics.TrackingId`* [String]
The tracking id provided by Google Analytics.

`CookieNotifier.Enabled` [true | **false**]
In accordance with the EU Cookie Law [2] we provide a notify banner at the bottom of the page that informs users that we use cookies. Set this to `false` to disable this banner.

`Germinate.AvailablePages`$^{\circ}$ [CSV]
Not all pages are useful for all instances of Germinate. List the names of those pages that should be available for this instance in a comma-separated fashion, e.g. `climate, megaEnvironments, gallery`.

`Germinate.ShowHomeOnLogin` [true | **false**]
If set to `true`, the login page will show the content of the home page as well. If set to `false`, the login page will only contain the text fields for the username and password.

`Germinate.Server.Logging.Enabled` [true | **false**]
If set to `true`, exceptions will be logged on the server side. Otherwise, they will only be forwarded to the client and the client decides what to do with them.

`Germinate.IsUnderMaintenance` [true | **false**]
If set to `true`, the web interface will be completely disabled, just showing a notification that the system is under maintenance.

`Germinate.IsReadOnly` [true | **false**]
If set to `true`, the web interface will not allow the user to perform changes to the database, i.e., the generation/modification of user-created content will be disabled.

`Germinate.ExternalDataFolder`* [Path]
If this property is set, Germinate will use the given path to look for files like images, data files and external applications. If the property is not set, Germinate will use the internal folders. This option can be useful if your data is huge and you don't want to include it in the generated war file, but rather want to store it in a different location on the server. Make sure that Germinate has read and write access to this folder.

`Germinate.HideIdColumns` [true | **false**]
: If set to `true`, the internal id column will be hidden from all tables.

`Germinate.Gallery.Images.Per.Page` [$x \in \mathbb{N}^+$; default: 16]
: This setting determines how many images are shown per page on the gallery page.

`Germinate.Gallery.Make.Thumbnails.Square` [true | **false**]
: If set to `true`, the automatically generated thumbnails of images will be square to keep a uniform look across all thumbnails. The default is `false` to keep it consistent with previous versions.

`Germinate.Template.CustomMenu`* [XML]
: This allows you to customize the main menu of Germinate. Please consult Section 3.1 for more details and an example.

`Germinate.Template.CategoricalColors`$^\circ$ [CSV(HEX)]
: A list of comma separated HEX color values (including the hash) that are used to color categories in some of the charts.

`Germinate.Template.GradientColors` [CSV(HEX); default: `#000000`, `#570000`, `#ff0000`, `#ffc800`, `#ffff00`, `#ffffff`]
: A list of comma separated HEX color values (including the hash) that are used for gradients (low to high).

`Germinate.Template.Social.ShowFacebook` [true | **false**]
: Set to true if a Facebook share button should appear on the site.

`Germinate.Template.Social.ShowTwitter` [true | **false**]
: Set to true if a Twitter share button should appear on the site.

`Germinate.Template.Social.ShowGooglePlus` [true | **false**]
: Set to true if a Google+ share button should appear on the site.

`Germinate.Template.UseToggleSwitches` [**true** | false]
: Set to true if you want to use toggle switches instead of two radio buttons whenever the user has to decide between a ”yes” and a ”no” option.

`Germinate.Template.Logo.Contains.Link` [true | **false**]
: Set to true if the main website logo SVG file contains links. In that case, Germinate will disable the default logo link to ”home” and prioritize the SVG-internal links.

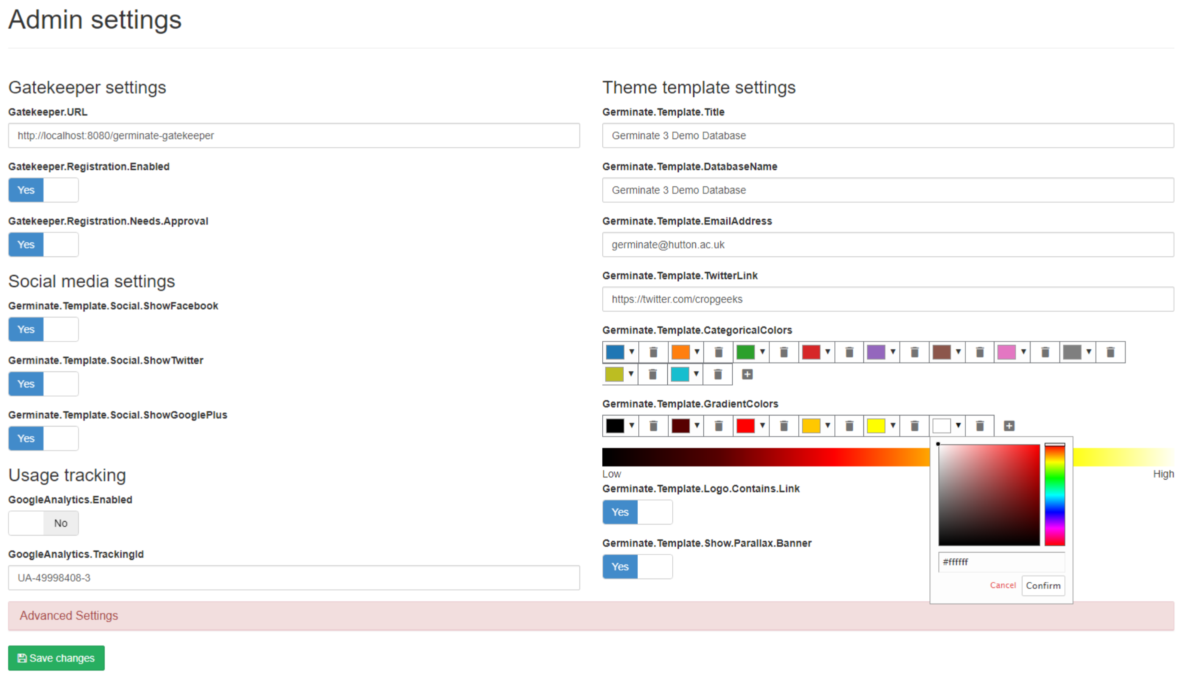
`Germinate.Template.Show.Parallax.Banner` [**true** | false]
: Determines if the parallax scrolling image banner is shown at the top of selected pages or not.

`Germinate.Template.EmailAddress` [E-Mail]
: The email address that will be displayed on the page as a contact address.

`Germinate.Template.Title` [String]
: The text to show in the browser title.

`Germinate.Template.DatabaseName` [String]
: The text to show in the ”featured banner” on the page.

`Germinate.Template.TwitterLink` [URL]
: The link to the associated Twitter account.



Admin settings

Gatekeeper settings

Gatekeeper.URL: http://localhost:8080/germinate-gatekeeper

Gatekeeper.Registration.Enabled: Yes

Gatekeeper.Registration.Needs.Approval: Yes

Social media settings

Germinate.Template.Social.ShowFacebook: Yes

Germinate.Template.Social.ShowTwitter: Yes

Germinate.Template.Social.ShowGooglePlus: Yes

Usage tracking

GoogleAnalytics.Enabled: No

GoogleAnalytics.TrackingId: UA-49998408-3

Theme template settings

Germinate.Template.Title: Germinate 3 Demo Database

Germinate.Template.DatabaseName: Germinate 3 Demo Database

Germinate.Template.EmailAddress: germinate@hutton.ac.uk

Germinate.Template.TwitterLink: https://twitter.com/cropgeeks

Germinate.Template.CategoricalColors

Germinate.Template.GradientColors

Germinate.Template.Logo.Contains.Link: Yes

Germinate.Template.Show.Parallax.Banner: Yes

#ffffff

Figure 3: The Germinate configuration page available to administrators.

`Path.Java` [String]
: The path to the Java installation. This is optional, since Germinate can get it from the JVM it's running in. However, if required, Germinate can be pointed to a different Java version and will use this to run all internal Java calls.

`Path.R`* [String]
: The path to the R installation.

Any change to these properties will automatically be picked up by Germinate, i.e. no reload of the web interface on Tomcat is required.

There is also a configuration page directly in Germinate. This page is only visible to administrators and can be reached by going to:

```
http://<Your web server>:8080/<project.name>/#admin-config
```

An example of this page can be seen in Figure 3.

3.1 Germinate Menu

Germinate uses a predefined menu structure by default. This default menu setup is based on what we think is a reasonable structure. However, even we are not infallible, so we decided to make the menu customizable.

We will now explain how you can define your own, custom menu structure and we will then give an example of how this can be used.

3.1.1 Structure

The menu of Germinate can be defined in XML format. We will explain this format using the example below:

```
<menu>
    <item key="home" icon="mdi-home-circle">
        <label key="en_GB">Home</label>
        <label key="de_DE">Home</label>
    </item>
    <item key="category.data">
        <label key="en_GB">Data</label>
        <label key="de_DE">Daten</label>
        <item key="browse-accessions">
            <label key="en_GB">Accessions</label>
            <label key="de_DE">Muster</label>
        </item>
        <item key="category.molecular">
            <label key="en_GB">Molecular data</label>
            <label key="de_DE">Molekulare Daten</label>
            <item key="genotype-datasets">
                <label key="en_GB">Genotypes</label>
                <label key="de_DE">Genotypen</label>
            </item>
            <item key="map-details">
                <label key="en_GB">Maps</label>
                <label key="de_DE">Molekulare Karten</label>
            </item>
        </item>
    </item>
    <item key="category.environment">
        <label key="en_GB">Environment</label>
        <label key="de_DE">Umwelt</label>
        <item key="geographic-search">
            <label key="en_GB">Geographic search</label>
            <label key="de_DE">Geografische Suche</label>
        </item>
    </item>
</menu>
```

This will result in the following menu structure:

British English:

- Home
- Data
  - Accessions
  - Molecular data
    - Genotypes
    - Maps
- Environment
  - Geographic search

German:

- Home
- Daten
  - Muster
  - Molekulare Daten
    - Genotypen
    - Genetische Karten
- Umwelt
  - Geographische Suche

We will now explain the individual elements of the XML file:

`menu` The root element of the XML file.

`item` This is used for any menu element. This can either be a link to an actual page, or a submenu.

`item key` If this item represents an actual Germinate Page (cf. Section 4.3) then this has to be the name of the page. If this item is a sub-menu, then this has to be a unique id.

`item icon` The icon that should be used for this menu item. It has to be one of the icons from `https://materialdesignicons.com/`. They are all prefixed with ”`mdi-`”. If no icon is provided, no icon will be shown.

`label` Labels are used as the text content of the menu item, i.e. what the user will see. Every item has to provide this information for ALL of the supported languages (cf. Section 6.3).

`label key` The locale of the supported language.

4 Features

In this section, we will highlight the main features of Germinate. This section will expand as we add new features.

4.1 Internationalization and Localization

Germinate fully supports internationalization for an unlimited number of languages. Every text that you can see on the web interface can be localized. The language used on start-up is chosen based on the browser settings, but the user can easily switch to a different language by selecting it from the combo box at the bottom of the page. Have a look at Section 6.2 for usage details.

4.2 News

The Germinate web interface contains a place holder to show the latest news of the project. On the database side, these news are stored in two tables and can be updated at any time. The web interface will check the availability of news and show the three latest entries at the bottom of the page. Changes on the database side will immediately be visible on the website.

4.3 Available pages

This section gives an overview of all the pages that are available for Germinate. Based on your requirements and the data that is available, you can decide which pages should actually be available on the web interface and hide all others. The set of available pages can be changed dynamically without having to re-deploy the application. This gives you the freedom to customize your Germinate experience just as you please.

4.3.1 About Germinate

Content
The about page of Germinate shows information about Germinate itself. This includes information about the developers and contact details.

Page name
`about-germinate`

Used data
None

4.3.2 About Project

Content
This page shows information about the project itself.

Page name
`about-project`

Used data
None

Customization You can change the content of this page by providing HTML to the `page.about.project.text` property.

4.3.3 Accessions Overview

Content
The overview page shows all the accessions contained in the Germinate database in a filterable table. The user can sort, filter and mark items. Below the table, there are tools to download accessions, their attributes and pedigree data to flat files.

Page name
`accession-overview`

Used data
biologicalstatus, collectingsources, countries, germinatebase, institutions, locations, locationtypes, mlsstatus, pedigreedefinitions, pedigreedescriptions, pedigreenotations, pedigrees, storage, storagedata, subtaxa, synonyms, synonymtypes, taxonomies

4.3.4 Accessions for Collecting Site

Content This page shows all accessions that have been collected at the given collecting site. It also allows the user to download the data in .kml format for Google Earth.

Page name
`accessions-for-collsite`

Used data
germinatebase, locations, locationtypes, countries

URL parameter
`collectingsiteId` The id of the collecting site.

4.3.5 Acknowledgements

Content
This page can be used to acknowledge collaborators, etc.

Page name
`acknowledgements`

Used data
None

Customization You can change the content of this page by providing HTML to the `page.acknowledgements.text` property.

4.3.6 Administrator Configuration

Content
This page allows page administrators to make modifications to the way Germinate operates and looks. Regular users cannot see this page.

Page name
`admin-config`

Used data
None

4.3.7 Allele Frequency Datasets

Content
All available allele frequency datasets are shown on this page. The user will be able to select which datasets to export.

Page name
`allele-freq-dataset`

Used data
allelefrequencydata, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes

4.3.8 Allele Frequency Export

Content
On this page, the user is able to select which data they want to export from the selected dataset.

Page name
`allele-freq-export`

Used data
allelefrequencydata, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes, groupmembers, groups, maps, mapdefinitions, mapfeaturetypes, markers, markertypes

URL parameter
`allelefreqDatasetIds` The id of the allele frequency dataset. Single selection only.

4.3.9 Allele Frequency Results

Content
On this page, the user can select the data binning scheme they want to apply to the allele frequency data. The final output files are shown on this page as well.

Page name
`allele-freq-results`

Used data
allelefrequencydata, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes, groupmembers, groups, grouptypes, maps, mapdefinitions, mapfeaturetypes, markers, markertypes

4.3.10 Climate Data

Content
The climate data page shows detailed information about climate information. The data is visualized in charts and a color-coded table. In addition, climate data map overlays are shown on a map.

Page name
`climate-data`

Used data

climatedata, climateoverlays, climates, countries, datasets, datasetpermissions, datasetstates, experiments, experimenttypes, germinatebase, groupmembers, groups, grouptypes, locations, locationtypes, units

4.3.11 Climate Datasets

Content

All available climate datasets are shown on this page. The user will be able to select which datasets to export.

Page name

`climate-dataset`

Used data

climatedata, climates, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes

4.3.12 Chemical Compounds

Content

This page shows a table with all the chemical compounds.

Page name

`compound-details`

Used data

analysismethod, compounddata, compounds, datasets, germinatebase, units

4.3.13 Chemical Compound Details

Content

The chemical compound details page shows information about a specific compound. This includes the compound data values across all accessions and visible datasets as well as that same data in the form of a chart. If images and external links are associated with a compound, these will be shown as well.

Page name

`compound-details`

Used data

analysismethod, compounddata, compounds, datasets, germinatebase, synonyms, synonymtypes, units

URL parameter

`compoundId` The id of the chemical compound.

4.3.14 Chemical Compound Datasets

Content

All available chemical compound datasets are shown on this page. The user will be able to select which datasets to export.

Page name

`compound-dataset`

Used data
compounddata, compounds, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes

4.3.15 Chemical Compound Data

Content
This page offers various different visualizations for the chemical compound data of the selected datasets. The data can also be downloaded into flat files.

Page name
`compound-data`

Used data
analysismethod, compounddata, compounds, datasets, germinatebase, synonyms, synonymtypes, units

URL parameter
`compoundDatasetIds` The ids of the chemical compound dataset.

4.3.16 Cookie

Content
EU law requires us to provide information about the cookies that Germinate uses and what we use this information for. Detailed information about our usage of cookies is available on this page.

Page name
`cookie`

Used data
None

4.3.17 Data Statistics

Content
The data statistics page visualizes aggregated information about the data contained in Germinate.

Page name
`data-stats`

Used data
Everything

4.3.18 Dataset Overview

Content
This page shows all the datasets within Germinate independent of their type in one table.

Page name
`dataset-overview`

Used data
allelefrequencydata, climatedata, climates, compounddata, compounds, datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes, phenotypedata, phenotypes, units

4.3.19 Experiment Details

Content
The experiment details page shows all the datasets contained in a single experiment.

Page name
`experiment-details`

Used data
datasets, datasetpermissions, datasetstates, experiments, experimenttypes

URL parameter
experimentId The id of the experiment.

4.3.20 Genotype Datasets

Content
All available genotype datasets are shown on this page. The user will be able to select which datasets to export.

Page name
`genotype-dataset`

Used data
datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes

4.3.21 Genotype Export

Content
On this page, the user is able to select which data they want to export from the selected dataset.

Page name
`genotype-export`

Used data
datasets, datasetstates, datasetpermissions, germinatebase, experiments, experimenttypes, groupmembers, groups, maps, mapdefinitions, mapfeaturetypes, markers, markertypes

URL parameter
`genotypeDatasetIds` The id of the genotype dataset. Single selection only.

4.3.22 Geographic Search

Content
The geographic search page allows the user to search for both accessions and locations based on a given query point or polygon.

Page name
`geographic-search`

Used data
countries, germinatebase, locations, locationtypes

4.3.23 Group Preview

Content
Germinate offers the functionality to define groups of accessions via an API and a dedicated page. Tools like CurlyWhirly can upload a file containing the identifier of accessions and Germinate will display a preview of these items in a table. Once the user is happy with the selection, they can create a new group based on the selection.

Page name
`group-preview`

Used data
germinatebase, groupmembers, groups, grouptypes

4.3.24 Groups

Content
The groups page shows an overview of all the accession, marker and location groups that are defined in Germinate. Depending on the permissions, the user can create new groups, delete old groups and modify the members of a group.

Page name
`groups`

Used data
countries, germinatebase, groupmembers, groups, grouptypes, locations, locationtypes, markers, markertypes

URL parameter
`groupId` The id of the group.

4.3.25 Image Gallery

Content
This page shows all the images that are defined grouped by type. By clicking on the button below each image the user will be able to navigate to the corresponding details page.

Page name
`image-gallery`

Used data
compounds, germinatebase, images, imagetypes

URL parameter
`latitude`, `longitude` The location of the point query.

4.3.26 Locations

Content
The locations page shows information about the location data. The locations are displayed on maps as well as visualized in different ways.

Page name
`locations`

Used data
countries, germinatebase, locations, locationtypes

4.3.27 Home

Content
The home page of Germinate. It shows aggregated statistics about accessions, groups, locations and markers.

Page name
`home`

Used data
countries, germinatebase, groupmembers, groups, grouptypes, locations, locationtypes, markers, markertypes

4.3.28 Institutions

Content
This page shows all the institutions associated with accessions held in Germinate.

Page name
`institutions`

Used data
countries, germinatebsae, institutions, locations, locationtypes

4.3.29 Maps

Content
The maps page shows an overview of all the maps. Selecting one of them will display a table of all the markers on this map along with their position on the chromosome.

Page name
`maps`

Used data
mapdefinitions, mapfeaturetypes, maps, markers, markertypes

URL parameter
`mapId` The id of a map. This will expand the details of this map.

4.3.30 Marked Items

Content
Marked item lists are a way of keeping track of items of interest. These items can be any accessions, markers or locations that the user is interested in. The marked items page shows all the items a user has marked so far. There are options to download the data or to create a group from the list if the user has sufficient permissions.

Page name
`marked-items`

Used data
germinatebase, groupmembers, groups, grouptypes, markers, markertypes, locations, locationtypes

4.3.31 Marker Details

Content
This page displays information about a single marker. It shows which datasets the marker is part of as well as which maps it is on. Additionally, a list of all known synonyms is shown.

Page name
`marker-details`

Used data
mapdefinitions, mapfeaturetypes, maps, markers, markertypes, synonyms, synonymtypes

URL parameter
`markerId` The id of the marker.

4.3.32 Mega Environments

TODO

4.3.33 News

Content
This page shows news about Germinate and the contained data. The news items are sorted by creation date, so the latest one is shown first.

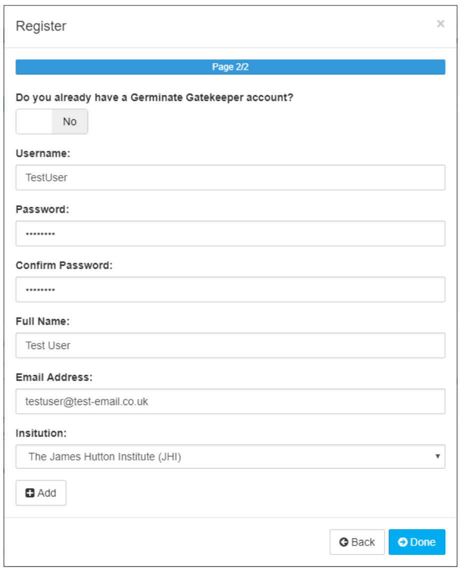
Page name
`news`

Used data
news, newstypes

4.4 User registration

Germinate can be used with and without authentication. This means that you can decide if you want to restrict access to your Germinate instance to registered users. If you decide to do so, you'll need people to be able to register. The registration form of Germinate (Figure 4a) provides a concise and easy way to register. If you choose to use a disclaimer that potential new users have to accept, this will appear before the form is visible. Once the user completes the form, they will either get access right away or you need to approve the new users manually. This is based on the setting of the property `Gatekeeper.Registration.Needs.Approval` (see Section 3 for help). If you choose to use the approval approach, you'll see the screen shown in Figure 4b after selecting "Approve users" in Gatekeeper. The user will be notified with your decision.

The registration agreement text the user has to agree to before signing up for your instance of Germinate is stored in the properties `page.registration.disclaimer.short.html` and `page.registration.disclaimer.long.html` in the `Text.properties` file. Translations of this text need to be placed in the appropriate internationalized version of this file.


Register ×

Page 2/2

Do you already have a Germinate Gatekeeper account?

No

Username:

TestUser

Password:

••••••••

Confirm Password:

••••••••

Full Name:

Test User

Email Address:

testuser@test-email.co.uk

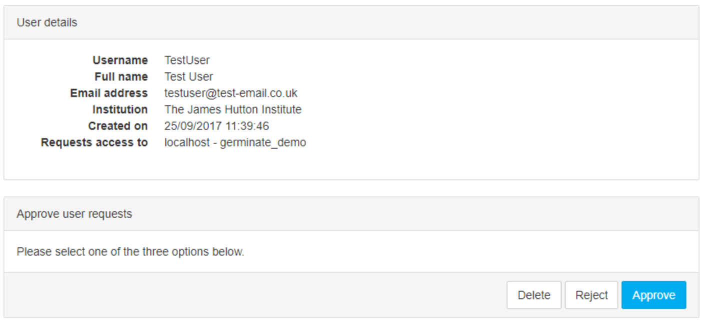
Insitution:

The James Hutton Institute (JHI)

Add

Back Done

(a) Registration form on the Germinate site


User details

Username	TestUser
Full name	Test User
Email address	testuser@test-email.co.uk
Institution	The James Hutton Institute
Created on	25/09/2017 11:39:46
Requests access to	localhost - germinate_demo

Approve user requests

Please select one of the three options below.

Delete Reject Approve

(b) Approval page on the Gatekeeper page

Figure 4: User registration

4.5 Dataset licenses

Sometimes it is required for individual datasets to be protected behind a license the user has to agree to before using the data. To accommodate this, we have introduced dataset licenses. Licenses are stored in the `licenses` table. The actual license content is stored in `licensedata` and linked to a `locale` so that you can have translations of your license content.

We keep track of which user accepted which license so that they are only prompted to accept new licenses. Make sure to remove associated `licenselogs` entries if you change the actual content of a license, because the user may originally have agreed to a different wording.

4.6 Security

As already mentioned earlier, Germinate is equipped with a secure login system[1]. This feature is completely optional, but it allows you to protect your data from any unauthorized access. In this section, we will explain how the security system works and we will show what is necessary to use it properly.

Figure 5 shows the authentication process of Germinate. The figure contains two major parts. The upper part visualizes the login process, which is initialized by the user entering his/her credentials. These are sent to the server alongside any session id that is still stored in the client from previous sessions. The server will then check the session id that is received. If the id is still valid, the server will store the session id and return it to the client, which, in turn, will create a cookie containing this id.

If the session id is invalid, the server will check if the username password combination is genuine. This is done by encrypting the password and checking it against the entry in the database. If this check fails, the user will be logged out. Otherwise, the server will create a new session id, store it in the HTTP session and return it to the client, which will create a new cookie using this id. The login process is now complete and both the server (via HTTP session) and the client (via cookie) know the current session id.

[1]Germinate is only as secure as the connection between client and server. Use a secure connection (TLS/SSL) to prevent password snooping.

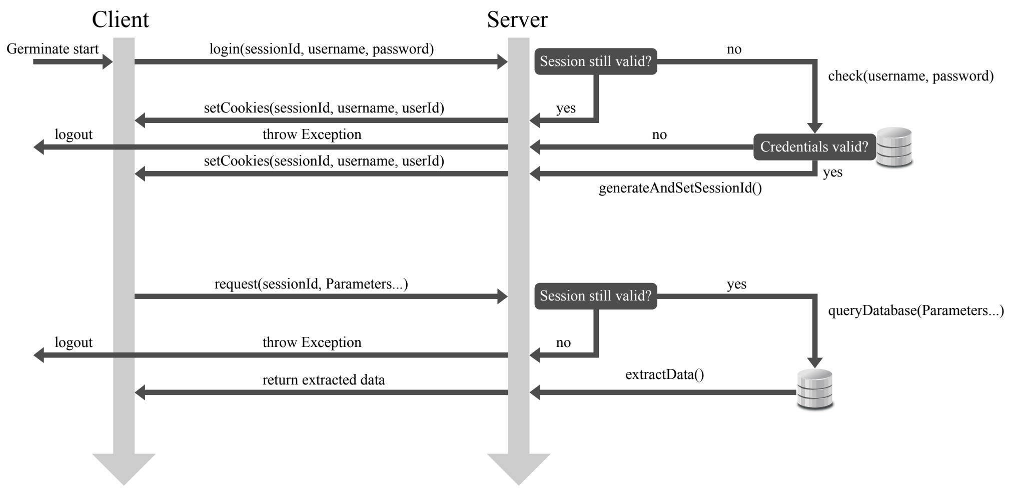

Client
Server
Germinate start
login(sessionId, username, password)
Session still valid?
no
check(username, password)
yes
setCookies(sessionId, username, userId)
logout
throw Exception
no
Credentials valid?
setCookies(sessionId, username, userId)
yes
generateAndSetSessionId()
request(sessionId, Parameters...)
Session still valid?
yes
queryDatabase(Parameters...)
logout
throw Exception
no
return extracted data
extractData()

Figure 5: Authentication procedure of Germinate

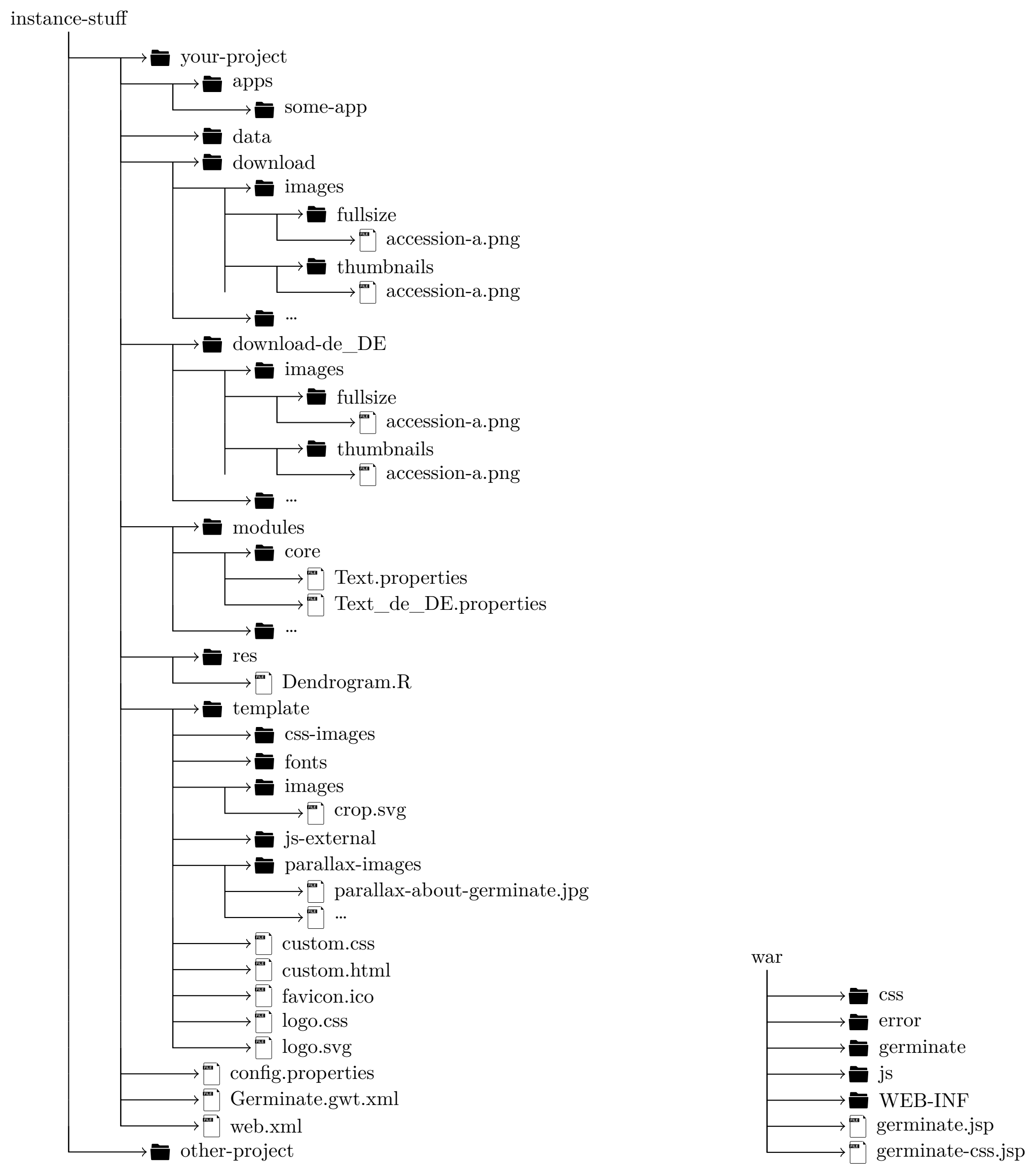
For each new request that is made from the client, it will need to send the session id as the payload, i.e. each remote procedure call (RPC) has to request the session id as a parameter to ensure the security of the communication. As a consequence if this, the server will receive the session id three times per request, namely via the HTTP session, via the cookie and via the payload. If all of these ids match up, the server can fulfil the client's request and return the data. However, if it fails, the user will be logged out.

As a final remark: Even if you currently do not want to enable the security feature, you should still write your code in a way that ensures it will work properly even when the security feature is enabled. Otherwise you might end up with a security leak.

5 Structure

In Section 5.1, we will explain the structure of the Germinate project. This includes the folder structure of the subversion project as well as the structure of the deployed product. Section 5.2 contains information about the structure of the database itself.

5.1 GitHub project



(a) Overall structure (b) Web structure

Figure 6: Available notifications

The actual subversion project of Germinate is structured as shown in Figure 6. Figure 6a shows the overall structure of the whole project whereas Figure 6b shows the structure of the files and folders related to the web side of things.

5.1.1 Project structure

Germinate is designed to allow custom look and feel for different projects. This means you are able to have multiple configurations of Germinate, each tailored specifically for the special needs of the individual projects. As an example we show the structure of the project ”your-project” which in reality would be named after your project.

There are several sub-directories. The ”apps” directory includes external applications that ship with Germinate. In this example, we include an application called ”some-app”. The ”data” directory contains raw data files that are used during the export processes. The ”download” directory contains all the data files that you want to provide as downloads on Germinate. Those can include images for the gallery, pdfs, etc. The ”download-de_DE” directory contains internationalized versions of the files found in ”download”. See Section 6.4 for further details. The ”modules” directory contains the internationalization property files for each module. In this example, there is only one module, namely the ”core” module which is the basic Germinate module. Internationalized text goes here. The ”res” folder can be used to host several files that are used as resources. The content of this folder is copied to the server and accessible from the server side code. In this example we have an R file that is used on the server to generate a dendrogram.

Finally, the ”template” folder contains custom styling files for your individual instance of Germinate. Those include custom logos, custom css, custom html, custom JavaScript libraries, custom parallax images, fonts, a favicon, etc.

The other files in your project folder are the most important ones. They define how Germinate works. We'll give a detailed description of each file below.

5.1.1.1 Config properties

`config.properties` is the file you have already read about in Section 3. It is used to define how to connect to the database, where to find Flapjack and R, which title to use for Germinate and so on.

5.1.1.2 Germinate xml

`Germinate.gwt.xml` is a very technical file. It should be left alone unless you have a good working knowledge of the Google Web Toolkit. There is, however, one thing in here that is easily to configure which are the supported languages.

An example of the internationalization properties can be found below:

```
<!-- Internationalizations -->
<extend-property name=”locale” values=”en_GB” />
<extend-property name=”locale” values=”de_DE” />
<set-property-fallback name=”locale” value=”en_GB” />
```

The `extend-property` entries list the supported language, or more exact, the locales. In this case, we support German and British English. You can add any locale you want to this file as long as you also provide a language properties file of the form `Text_xx_XX.properties`.

The `set-property-fallback` entry defines the default locale as well as the locale that is used if an unsupported locale is requested. It has to be one of the previously defined locales. In this case we use British English.

5.1.1.3 Web xml

Every Java web developer will be familiar with this file. We will give a short explanation of the content here.

```
<display-name>{\germinate} Template Database</display-name>
```

```
<!-- LISTENERS -->
<listener>
    <listener-class>jhi.germinate.server.util.ApplicationListener</listener-class>
</listener>

<!-- Default page to serve -->
<welcome-file-list>
    <welcome-file>germinate.jsp</welcome-file>
</welcome-file-list>

<!-- CUSTOM ERROR PAGES -->
<error-page>
    <error-code>404</error-code>
    <location>/error/error.jsp</location>
</error-page>
```

The code above shows part of the template file. `welcome-file` is the default file that is served to the browser on the first visit. The `error-page` entries define the error pages that Germinate will take care of. All other error pages are handled by Tomcat instead.

5.1.2 Web structure

The most important file in this structure is `germinate.jsp`. This file contains the basic web skeleton of the website, i.e. the overall structure. In this file, you should define external JavaScript files and external css files you want to load in addition to the already contained ones. `germinate-css.jsp` contains the custom Germinate style sheet. Most of what you find in this file takes care of how the content of the page looks. To change the styling of the overall template (not the content), use this file: `css/style-css.jsp`.

As you might already have notices, both files are not typical `.css` files, but rather `.jsp` files. However, the first line in all of these files lets the browser know, that in fact they are css stylesheets. The reason for using jsp instead is based on the fact, that we have access to the `PropertyReader` in the jsp files. This allows us to get properties from the server and change the stylesheets before they are sent to the browser.

We use this feature to get the so called ”highlight color” from the properties file. This color is used for most of the header elements on the web site as well as several other things including links. If we change this color in the properties file, the stylesheets will pick it up and use this color from now on. This centralized mechanism reduces errors and makes it easier to change things quickly.

As mentioned earlier, you can simply add external css and js files to Germinate. There are predestined folders for these here as well. The `germinate` folder is created by Eclipse when you compile the project. It contains all the JavaScript source files that are generated and served to the browser. You don’t need to worry about this folder at all. The `error` folder contains resources for the error pages. At the moment Germinate provides humorous error pages for the most common errors: 404, 403 and 401. Feel free to add further error pages.

Finally, the `WEB-INF` folder should look familiar to every web developer. It contains the libraries (in form of `.jar` files that are necessary to run Germinate on the server as well as the compiled Java source.

5.2 Database

We moved the documentation of the database to an online resource. You can find an overview of all the tables with detailed descriptions at the following location:

`https://ics.hutton.ac.uk/resources/germinate/model/germinate_3_4_0/`

6 Customization

Possible changes to Germinate may range from basic formatting changes to the modification of existing content. The following sections contain detailed examples for the most common scenarios of extension. Each example consists of the necessary code and accompanying explanations.

6.1 Customizing the Germinate theme

Germinate uses Bootstrap [3] for its web interface. Bootstrap is one of the most popular open-source web front-end frameworks. It allows us to design Germinate in a consistent and user friendly way without a lot of customization.

Another advantage of Bootstrap is that you can use custom themes. The theme Germinate uses includes some minor changes from the default Bootstrap theme. If you would like to make changes to the way Germinate looks, you can easily customize the existing theme by uploading our custom `config.json` (located in the root directory of the source code) file to:

`https://getbootstrap.com/docs/3.3/customize/`

You can customize things like colors, fonts, font sizes and many more settings. Once you are happy with your changes, download the resulting files and include the content of `bootstrap.css` into the `custom.css` file that is part of your instance configuration files.

6.2 Internationalizing your pages

As already mentioned in Section 4.1, Germinate supports internationalization for as many languages as you want. Internationalizing your custom code is really easy:

1. Define a method that returns your localized text in `jhi.germinate.client.i18n.Text`.
2. Add the localized values for this new variable to the `.properties` files located at `instance-stuff/<project.name>/modules/core/`. The file naming convention is: `Text_<Language>_<Region>.properties`.

A list of supported locales is available at [4].

As an example, we will now add a new localized String to the application. To get the localized text, we used the method `Text.LANG.menuData()` which returned ”data” for the English version of the page. Now let’s assume we want to add a menu item that has the text ”my fancy page”. The first thing we have to do is create a new method in the interface `jhi.germinate.client.i18n.Text`:

```
String menuMyFanyPage();
```

As we can see, this method will return a `String` which we can use in our menu. You can regard this method as a template for the localization.

The second step is to provide a translation of this text for each of the supported languages:

```
menuMyFanyPage=my fancy page
```

Add this to the files mentioned in the second bullet point above.

GWT will substitute the template calls with the appropriate localized text during compile time.

Table 1: Available plural form rules in GWT

Category	Rules	Examples
zero	x is 0	0
one	x is 1	1
two	x is 2	2
few	$x \bmod 100 \in \{3, ..., 10\}$	3-10, 103-110, ...
many	$x \bmod 100 \in \{11, ..., 99\}$	11-99, 111-199, ...
other	Everything else	100-102, 200-202, 11.68, ...

6.2.1 Parametrization and Plural Forms

Instead of just returning static text from the properties files, Germinate is able to substitute place holders with parameters allowing for a dynamic usage of localization. One specific case of parameter usage is the case of plural forms. The text you want to display might differ based on the number of menus it should represent. As an example, consider the sentence: ”I can see 3 trees”. The sentence changes for just one tree: ”I can see a tree”. The method returning this text could look like this:

```
String iSeeTrees(@PluralCount(DefaultRule_en.class) int number);
```

The annotation `@PluralCount` tells the method that the parameter called `number` is a countable variable. The associated properties entries could look like this:

```
iSeeTrees=I can see {0} trees.
iSeeTrees[one]=I can see a tree.
iSeeTrees[many]=I can see many trees.
```

As you can see, the plural form is handled by the first line, while the singular form is handled by the second line. A list of available plural forms can be seen in Table 1. For more information, please refer to [5].

Another case of parametrization is simple substitution. As an example we will create a method returning a welcome message for the user and the day of the week.

```
String welcomeMessage(String username, String day);
```

The method is defined just the way a normal method taking two parameters would be defined. The localized entry in the properties file could look like this:

```
welcomeMessage=Hello {0}. Welcome to {\germinate}. Today is {1}.
```

Calling `welcomeMessage("Joe Bloggs", "Wednesday");` will result in ”Hello Joe Bloggs. Welcome to Germinate. Today is Wednesday”.

6.2.2 Notes

- It is possible to include HTML tags in the localized strings. Consequently, you can change the font style (size, bold, italic,...), add hyperlinks, include images, etc.
- Use two single quotes instead of just one (e.g. ”It”s done” instead of ”It’s done”).
- The encoding of the properties files has to be UTF-8.

6.3 Adding a new language

We have seen how to add content to existing languages in the previous section. Now we will show how to add a completely new language to the application. To ensure neutrality, we selected

Switzerland and will use the language of Swiss German. The locale id of Swiss German is `de_CH`. The first part represents the language (de = Deutsch which is German for ”German”) and the second part represents the country (CH = Confoederatio Helvetica which is Latin for ”Swiss Confederation”).

Navigate to your configuration folder located at `instance-stuff/<project.name>` and create a new file called `Text_de_CH.properties`. Now copy the whole content of one of the fully translated other languages into this file and substitute all of the text with its appropriate translation.

Next, you'll need to make a change to this file: `war/css/language-selector-css.jsp`. Search for the line containing the 2-digit country code of the country associated with the new language. In our example, that would be `CH` and the line looks something like this:

```
span.country.ch { background-position: 0px -1440px; }
```

Add a new line with your new locale before this line. Mind the comma:

```
.language-selector span.de_CH,
span.country.ch { background-position: 0px -1440px; }
```

Finally, open the file `instance-stuff/<project.name>/Germinate.gwt.xml` and add this line:

```
<extend-property name="locale" values="de_CH" />
```

After compiling and deploying the project, you should be able to select the new localization from the language selector at the top of the page.

6.3.1 Notes

- The default locale is determined by the browser settings. A fall-back solution can be specified by this entry in the `Germinate.gwt.xml` file:

  ```
  <set-property-fallback name="locale" value="en_GB" />
  ```

- Not all localization files have to be complete. If translations are missing, they will be substituted by their respective value from the default locale.

6.4 Internationalized files and images

In some cases it may be necessary to supply a download file or image in more than just one language. We decided to adopt a concept called *resource qualifiers*. This concept is also used in the popular Android platform [6].

The main idea of this concept is to provide resources in alternate forms which will be used under certain circumstances. On Android one example of a resource qualifier are language and region. Valid examples are *de* for German resources or *en-rGB* for British English.

The advantage of this concept is that you only have to copy your files/images to the appropriate folder and the system will choose the correct file based on the current configuration. If a file does not exist in the folder for the current configuration, the system will fall back to the default file.

We adopted this idea for Germinate. We allow internationalization of files in the directories ”download”, ”data”, and ”res” (see Section 5 for details). To provide internationalized versions of your files, you need to create a new directory of the type you want to extend (e.g. ”download”) and then append the locale separated by a dash. In the previous sections we used the local `de_CH` as an example. The internationalized download directory for this locale would have to be named `download-de_CH`. The structure within this directory has to be identical to the base-directory

(”download”). Please note, that this concept only works if the files in the different directories have the same file name. Otherwise, Germinate won’t be able to find them.

The result for the user is a seamless internationalization of all the content within Germinate. When changing the locale on the web interface, the server backend will serve the internationalized files for this locale, if available, and fall back to the default locale otherwise.

6.4.1 Passing parameters via the URL

When Germinate is first loaded by the browser, it will parse the given URL parameters and save the respective parameters in the `ParameterStores`. As a result, you can create URLs that take the user to a specific page for a specific combination of parameters. As an example, the URL

```
http://<your_server>:8080/<project.name>/?accessionId=1#passport
```

will take you to the passport page of Germinate showing the accession with id 1. Alternatively, the URL

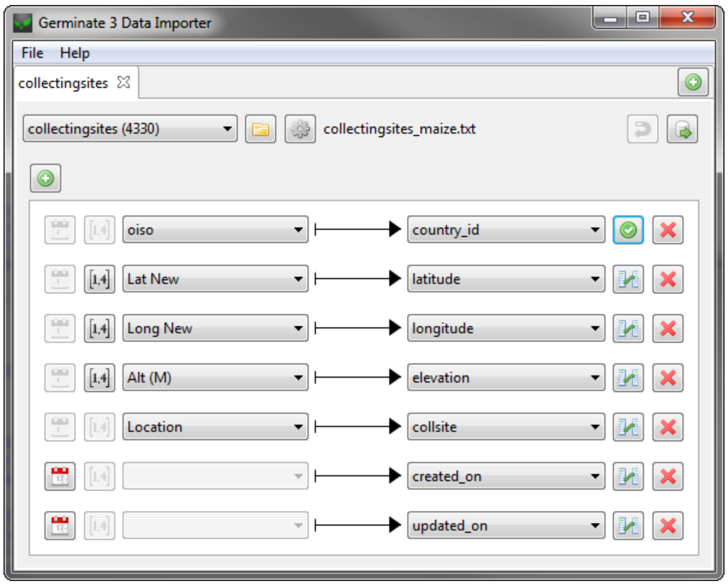
```
http://<your_server>:8080/<project.name>/?searchString=Baz#search
```

will show the search results for the given search string. Multiple parameters can be specified by combining them with an ampersand (&).

For security reasons, we only save valid parameters, i.e. parameters that are part of the enum `Parameter`. Moreover, it is very important that the parameters are located **before** the fragment identifier (#).

6.4.2 Custom Bootstrap Theme

TODO



oiso	country_id
Lat New	latitude
Long New	longitude
Alt (M)	elevation
Location	collsite
	created_on
	updated_on

Figure 7: Germinate Daim - Column mapping

7 Data Import

We realize that the task of importing data into a database as complex as Germinate may be overwhelming at first. This is why we try to come up with tools that make this process simpler and faster. In this section, we will highlight some of the possible ways you might want to use to import your data into the Germinate database.

7.1 Germinate Daim

There are several ways to import data into Germinate. Some people may prefer to write their own scripts to import the data whereas others may prefer database management tools like *Navicat* [7] or *MySQL Workbench* [8].

However, we believe that we can make the import process even easier and so we developed a Java based desktop application to aid in the upload of experimental datasets into Germinate. The aim of this tool is to simplify the process of entering data into the system. *Germinate Daim* allows permitted MySQL system users to upload data in text file format into a specified Germinate installation. We have also included tools to revert the most recent change should any problems arise in the data upload process.

Germinate Daim can be downloaded here:

`http://ics.hutton.ac.uk/germinate-daim/`

Figure 7 shows the main GUI of G3DI with an example mapping. The current tab is used to import data from a text file with the name `collectingsites_maize.txt` into the database table `collectingsites`. Each row represents a mapping between column of the input file and column of the database. Germinate Daim is aware of the data types of the database columns and provides functionality to accommodate dates, number formats and foreign keys. The first row of the mapping maps `oiso` (which is a 3-digit country code) to `country_id`. Germinate Daim knows that `country_id` is a foreign key column and requests the user to define a lookup table and column. This lookup is used to find the foreign id based on the value in the `oiso` column.

As an example, let's assume the lookup table is `countries` and the lookup column is `country_code3` and the value in the column `oiso` is "GBR". Germinate Daim will then query the database and determine the id of the country with the 3-digit code "GBR" (which is the United Kingdom) and use this id instead of the actual value.

In addition you can define a number format (or to be exact, a locale) for decimal columns. This format is used to parse your input data before inserting it into the database.

Finally, you can define date formats for columns that represent dates. As an example, let's say your input data is in the form `dd/MM/yyyy` (2-digit day, 2-digit month, 4-digit year). Germinate Daim will then parse your dates before inserting them into the database. This is necessary, since the database uses its own date format which may differ from the format you used for your data.

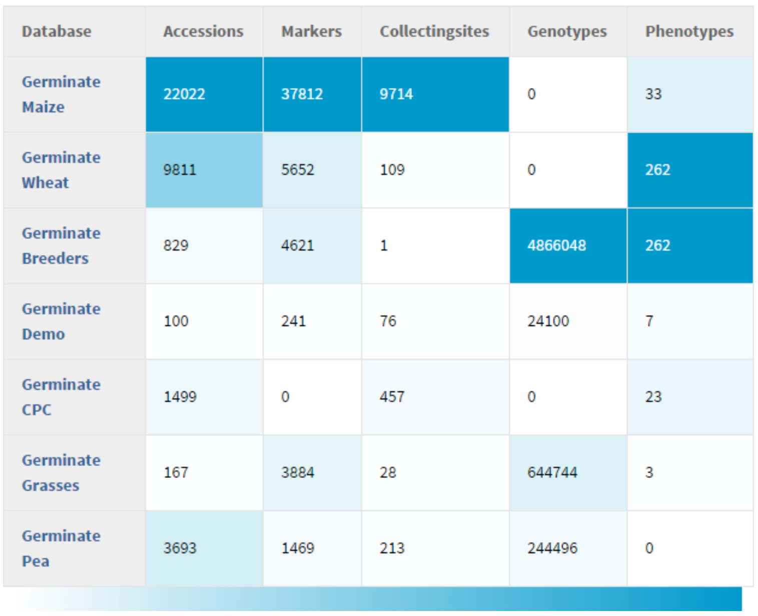
After you imported your data, you might realize that your mapping was incorrect. Germinate Daim can undo the most recent import step with one click of a button, to allow you to do your work without worrying that you might mess up the database. If you want to be on the safe side, you can still take a backup of the whole database with your favourite database management tool.

7.2 Germinate Data Templates

Over the years we have learned that people really like Microsoft Excel and that most of the data is stored in Excel spreadsheet before it goes into Germinate. To make the best of this fact, we decided to design Excel data templates that specify exactly how your data needs to be formatted for Germinate to import it automatically.

The "datatemplates" folder in the Germinate source code contains templates for genotypic data, genetic maps, germplasm passport data, pedigree information and trials data. Each of the files contains detailed information about how we require the data to be formatted.

Explain command line tools



Database	Accessions	Markers	Collectingsites	Genotypes	Phenotypes
Germinate Maize	22022	37812	9714	0	33
Germinate Wheat	9811	5652	109	0	262
Germinate Breeders	829	4621	1	4866048	262
Germinate Demo	100	241	76	24100	7
Germinate CPC	1499	0	457	0	23
Germinate Grasses	167	3884	28	644744	3
Germinate Pea	3693	1469	213	244496	0

Figure 8: Statistics overview table

8 Statistics

We maintain a Germinate statistics page on our server. It shows an overview of all Germinate instances that we currently maintain. It's available at:

`http://ics.hutton.ac.uk/germinate/germinate-statistics`

Figure 8 shows one of the visualizations we provide. We're planning to add more visualizations in the future. Now you might think: "What does that have to do with me?".

The answer is simple: We can add your instance of Germinate to our statistics website. That way, people browsing our website may stumble across your site which can broaden your potential target audience. In addition, access to statistical data from more Germinate instances allows us to improve Germinate and the statistics page. So it's basically a win-win situation.

Just in case you are wondering what kind of data we would collect from your Germinate instance, let me go into detail here: When you start Germinate on your server, it creates a couple of views. These views contain the aggregated data. You can examine these views with your favourite MySQL tool.

The way we can access this information depends on if your instance of Germinate has authentication enabled or not. If not, then there is nothing to do, we can invoke a service on your server using Germinate's API and pull back the aggregated data. If authentication is enabled, you would have to create a new user using Germinate Gatekeeper and send us the username and password. This user can be a regular user and does not have to be an administrator. We would then store the user credentials in our database and run a query against your views once a day. Consequently, the data we show on our statistics page is always up to date.

If you are interested and want us to add your Germinate instance to our statistics page, or if you have any questions, drop us an email: `germinate@hutton.ac.uk`.

9 Troubleshooting

In this section we talk about issues that you may encounter when working with Germinate.

9.1 Germinate links in download files are not HTTPS

Files downloaded from Germinate sometimes contain links back to certain pages back on Germinate. This is used so that external tools can come back to Germinate for additional information.

In case you're running Germinate over HTTPS (which you should) and are also using a reverse proxy to forward requests to Tomcat, it might happen that these links don't point to `https://your-server.com/germinate` but rather to `http://your-server.com/germinate`. In most cases, this will be fine as the user will be redirected to HTTPS and won't even notice that this happens. Some external tools might not be able to handle this redirect though.

The problem here is that Tomcat doesn't know that the request came to an HTTPS resource originally (before it got proxied) and Germinate therefore doesn't have sufficient information to generate the links correctly.

To fix this, open the `server.xml` file under Tomcat's `conf` folder and add the following `Connector` to the `Server`:

```
<Connector port="8081" protocol="HTTP/1.1"
    connectionTimeout="20000"
    proxyName="<proxy server name, e.g. baz.hutton.ac.uk>"
    proxyPort="443"
    scheme="https"
    secure="true"/>
```

In this case, `proxyName` is the name of the proxy server. In addition, you need to change the "proxy reverse" settings of Apache. Open the file containing your configuration and identify the entries responsible for Germinate. They may look something like this:

```
ProxyPass           /germinate-template http://tomcat-server:8080/germinate-template
ProxyPassReverse    /germinate-template http://tomcat-server:8080/germinate-template
```

Change the port from 8080 to 8081 (as configured on the `server.xml`) so that it looks like this:

```
ProxyPass           /germinate-template http://tomcat-server:8081/germinate-template
ProxyPassReverse    /germinate-template http://tomcat-server:8081/germinate-template
```

Restart both Apache and Tomcat for this fix to work.

10 Funding

Germinate 3 was developed with CIMMYT [9] as part of the MasAgro Seeds of Discovery Project [10].

11 Technologies

Germinate uses third-party software libraries to extend its functionality. The following sections will list these libraries and explain their purpose:

11.1 JavaScript libraries

Bootstrap Notify
We use Bootstrap Notify to display notifications on the website.
`https://github.com/mouse0270/bootstrap-notify`

Bootstrap Switch
Bootstrap Switch makes it easy to use switch or toggle buttons on the web interface.
`https://github.com/Bttstrp/bootstrap-switch`.

CookieCuttr
Because of the EU "Cookie Law", websites have to let the user know when they're using cookies. CookieCuttr displays a banner notifying the user of just that.
`https://github.com/weare2ndfloor/cookieCuttr`

D3.js
We us D3.js to generate dynamic and interactive data visualizations. Almost all charts in Germinate are created using this library.
`https://github.com/mbostock/d3`

d3Pie
This is a plugin for D3.js that we use to easily create pie charts.
`https://github.com/benkeen/d3pie`

D3.tip
This is a plugin for D3.js that allows easy creation of tooltips for SVG elements.
`https://github.com/Caged/d3-tip`

FancyBox
FancyBox is used to create popups of images and text.
`https://github.com/fancyapps/fancyBox`

html2canvas
html2canvas is used to convert the d3.js chart legend to an image and download it.
`https://github.com/niklasvh/html2canvas`

jQuery
jQuery is the most popular JavaScript library out there. It allows easy selection, traversal and manipulation of DOM elements.
`https://github.com/jquery/jquery`

lasso.js
This is a plugin for D3.js that allows freehand drawing of selections within D3.js charts.
`https://github.com/skokenes/D3-Lasso-Plugin`

Leaflet
Leaflet is an open-source JavaScript library for mobile-friendly interactive maps. We use it throughout Germinate to display data on geographic maps.
`https://github.com/Leaflet/Leaflet`

prettify.js
Prettify is used to make code on website more pretty, e.g. it adds basic syntax highlighting. The main use of this library is for the SQL debug messages that appear when Germinate is run in debug mode.
`https://github.com/google/code-prettify`

saveSvgAsPng.js
This library allows the user to save a SVG image in the browser (usually generated by D3.js) to a PNG file on their computer.
`https://github.com/exupero/saveSvgAsPng`

Spectrum
Spectrum allows us to easily add color-pickers to the web-interface. HTML5 introduced the color-type input element, but many current browsers still don't support it. We also always have to consider users with outdated browsers, so unfortunately we have to fall back to a third-party library for this purpose.
`https://github.com/bgrins/spectrum`

11.2 Web frameworks

Bootstrap
Bootstrap is one of the most popular web front-end frameworks out there. It provides HTML- and CSS-based templates for interface components and layouts alongside JavaScript extensions for additional functionality.
`https://github.com/twbs/bootstrap`

11.3 Font frameworks

Font Awesome
This is a truly awesome framework containing loads of pictographic icons that we use throughout Germinate. They are fully scalable which makes them a perfect replacement for raster images.
`https://github.com/FortAwesome/Font-Awesome`

Material Design Icons
Material Design Icons is an icon set inspired by Google's Material Design guidelines. We use it in addition to Font Awesome within Germinate.
`https://github.com/Templarian/MaterialDesign`

11.4 Java libraries

Apache Commons CLI
We use this library to easily handle command line parameters for out data import code.
`https://github.com/apache/commons-cli`

Apache Commons FileUpload
This library is used to make the upload of files from the browser to the server easier.
`https://github.com/apache/commons-fileupload`

Apache Commons IO
This is a widely used utility library.
`https://github.com/apache/commons-io`

Apache Commons Logging
A popular logging library.
`https://github.com/apache/commons-logging`

Apache HttpComponents Client
This library is used to easily communicate with Gatekeeper via HTTP requests.
`https://github.com/apache/httpclient`

Apache HttpComponents Core
Required by Apache HttpComponents Client.
`https://github.com/apache/httpcore`

Flapjack
Flapjack is a graphical genotype viewer developed at The James Hutton Institute. Germinate uses it on the server side to export genotypic data and allele frequency data.
`https://ics.hutton.ac.uk/flapjack`

Flyway
Flyway is a database migration tool that Germinate uses to update its database between releases.
`https://flywaydb.org`

GWT
GWT is the main building stone of Germinate. It's the web development framework we chose to use.
`https://github.com/gwtproject/gwt`

GWT-Charts
This is a GWT implementation of the JavaScript Google Charts API.
`https://code.google.com/p/gwt-charts`

GwtQuery
We use this library to use jQuery-like code in Java.
`https://github.com/ArcBees/gwtquery`

Intro.js
Intro.js is used to display interactive introduction tours that guide the user through a number of steps and explains certain parts of the web interface.
`https://github.com/usablica/intro.js/`

jBCrypt
jBCrypt is a Java implementation of the OpenBSD's Blowfish password hashing algorithm.
`https://github.com/jeremyh/jBCrypt`

JAK
This "Java API for KML" is used when exporting geographic information to KML format.
`https://github.com/micromata/javaapiforkml`

JAXB
Required by JAK.
`https://jaxb.java.net/`

MySQL Connector/J
This is a library that allows us to easily communicate with a MySQL database from Java code.
`https://github.com/mysql/mysql-connector-j`

SimpleXML
We use SimpleXML to parse the custom menu of Germinate easily.
`https://github.com/ngallagher/simplexml`

Thumbnailator
This is a utility library used to create thumbnails of images. Whenever a new image is copied to the full-size image folder of Germinate, we will use this library to automatically generate a thumbnail for it.
`https://github.com/coobird/thumbnailator`

References

[1] I. Milne, P. Shaw, G. Stephen, M. Bayer, L. Cardle, W. T. Thomas, A. J. Flavell, and D. Marshall. Flapjack - graphical genotype visualization. *Bioinformatics*, 26(24):3133–3134, 2010.

[2] The EU cookie law (e-Privacy Directive). http://ico.org.uk/for_organisations/privacy_and_electronic_communications/the_guide/cookies. Last accessed on 2014-08-05.

[3] Bootstrap. http://getbootstrap.com/, 2011. Last accessed on 2017-09-28.

[4] JDK 7 and JRE 7 Supported Locales. http://www.oracle.com/technetwork/java/javase/javase7locales-334809.html. Last accessed on 2013-09-20.

[5] I18n Plural Forms. http://www.gwtproject.org/doc/latest/DevGuideI18nPluralForms.html. Last accessed on 2013-09-23.

[6] Android Resource Qualifiers. http://developer.android.com/guide/topics/resources/providing-resources.html#AlternativeResources. Last accessed on 2014-09-07.

[7] Navicat. http://www.navicat.com/, 2002. Last accessed on 2014-07-23.

[8] MySQL Workbench. http://mysqlworkbench.org/, 2005. Last accessed on 2014-07-23.

[9] CIMMYT: International Maize and Wheat Improvement Center. http://www.cimmyt.org/. Last accessed on 2014-06-26.

[10] Seeds of Discovery: Unlocking the genetic potential of maize and wheat. http://seedsofdiscovery.org/, 2012. Last accessed on 2014-06-26.