Installation Guide

for

52°North terrainServer

version 1.0

Revision History

Revision Number	Date of Publication	Author(s)	Change Description
1.0	2012, Dec 06	Benno Schmidt	initialization, general description of setting up the software
1.01	2013, Jan 27	Benno Schmidt	minor changes
1.02	2013, Jan 29	Ann Hitchcock, Benno Schmidt	minor changes

Editor

Benno Schmidt
Bochum University of Applied Sciences
Department of Geodesy
Lennershofstr. 140,
44801 Bochum, Germany
Email: benno.schmidt@hs-bochum.de

License

Contact: Andreas Wytzisk
52°North Initiative for Geospatial Open Source Software GmbH,
Martin-Luther-King-Weg 24,
48155 Muenster, Germany, info@52north.org

For more information, contact:
52°North Initiative for Geospatial Open Source Software GmbH
Martin-Luther-King-Weg 24
48155 Münster, Germany
http://52north.org

Table of Contents

1 Introduction

1.1 Basic terrainServer functionality

The 52°North terrainServer is a Web application providing access to elevation information, in other words, z-value information is processed. In addition to topographic elevations, these values might represent other thematic contents, such as sea depth values, ground-water levels, pollutant concentrations, meteorological parameters, etc. For the sake of simplicity, this document will refer to elevation values only.

Perspective map views can be delivered. Access to elevation data, 3D scene representations, and rendered images occurs via HTTP-Get services and well-specified service interfaces. Thus, functionality provided by the 52°North terrainServer can be easily integrated into existing spatial information infrastructures.

52°North terrainServer provides various services:

1. The *WebTerrainServlet*, which offers
    - a Web Terrain Service (WTS) implementation to generate perspective views, e.g. a rendered PNG or JPEG image showing a static 3D view of the Earth's surface;
2. the *ProfileServlet*, which offers
    - a simple HTTP-Get service to query an elevation value for a specified position (x, y), as well as
    - a service to derive cross-sections from the underlying elevation data;
3. and the *DEMServlet*,
    - which implements a Web 3D Scene Service (W3DS) prototype delivering dynamic 3D scene descriptions, e.g. as VRML or X3D document, as well as
    - services delivering elevation models encoded in selectable ASCII data formats.

It is possible to install all of the services mentioned above or just single services.

This document describes the installation and configuration procedure of the terrainServer application as downloaded from the 52°North web page.

1.2 Prerequisites

- To run the terrainServer, a suitable Servlet environment is required; see paragraph 1.3.
- Elevation data (respectively other z-value data) is required and must be given as tiled elevation grids files. See chapter 2 for details.
- If you want to set up a Web Terrain Service, the POV-Ray rendering engine must be installed; see chapter 6.

1.3 System requirements

1.3.1 Servlet container

The software consists of various "Servlets", i.e. server side Java applications. To run the software, a suitable Servlet environment is required. It is assumed that Apache Tomcat 5.5, 6, or 7 is used as Servlet engine. Other Servlet containers should be usable as well.

This document assumes the use of Apache Tomcat as Servlet container and presumes that the user is familiar with the Tomcat software.

1.3.2 Java development kit (JDK)

The Servlet engine requires a Java Development Kit (JDK). The 52°North terrainServer supports JDK 1.5+.

If using Apache Tomcat, please refer to http://tomcat.apache.org/whichversion.html to find out which JDK is suitable for your needs!

1.4 About the downloaded binary

The terrainServer binary currently available from the 52°North web page is a pre-compiled web application packaged as a war archive.

The file name is: 52n-v3d-terrainServer-<version>.war

The binary file is a zip archive having the file extension .war. To configure this file, unpack it, apply the proposed changes in this document and add it to your Servlet container.

1.5 Internal file structure

The binary distribution contains the following set of folders:

META-INF/	Contains context information and information about the build process (for developers only)
WEB-INF/	
web.xml	Servlet container deployment descriptor. Note that this is the Web application's main configuration file, see explanations below.
classes/	Contains logger configuration file and Java implementation classes (Servlets)
ErrorMessages.asc	File containing human readable error messages
log4j.xml	Logger configuration file, see paragraph 4.2.
tiledef.properties	Tile definition file
conf/	Contains the services' GetCapabilities responses (as plain XML files); to be adapted to your specific environment.
lib/	Set of internal and third party libraries (JAR files)
THIRD_PARTY_LICENSES.txt	List of all third party libraries and licenses used

This folder will be referenced as $TERRAIN_SERVER_HOME in the next chapters.

Note: With UNIX or Linux, a forward slash (/) instead of backslash (\) has to be used as separator. For the sake of ease, we use the slash (UNIX notation).

1.6 Quick start installation

The installation procedure will be explained in detail in the next chapters. For those experienced users who just want to start quickly, here is a quick start guide:

Step 1. JDK installation (recommended: JDK 1.6)

Step 2. Servlet container installation (recommended: Apache Tomcat 6.0)

Step 3. Unzip the terrainServer WAR file.

Step 4. Provide your elevation grid files in a folder $DEM_DATA outside your Web application.

Step 5. Edit the $TERRAIN_SERVER_HOME/web.xml file.

Hint: For a quick start, it is often sufficient to configure just these two parameters:

SourceGridPath must point to $DEM_DATA

WorkingDirectory must point to your working directory, e.g. /temp/terrainServer.

Step 6. POV-Ray installation (recommended: POV-Ray 3.6). Note that you can ignore this, if you do not want to provide the WebTerrainServlet!

Step 7. Provide a terrainServer context file.

Step 8. Start your Servlet engine and try out your first request!

2 Elevation Data Preparation

2.1 Overview

To run the 52°North terrainServer, proper elevation information is needed.

- To date, the terrainServer supports ArcInfo ASCII grid files as elevation data source only. This is a widely used format with a simple structure fulfilling the terrainServer's needs.
- The 52°North terrainServer uses a tiling mechanism to access elevation information, thus suitable grid tiles have to be provided (for every coordinate reference system that shall be supported):
    - The covered region is divided into tiles of a fixed size.
    - Each tile's geometric extent will be determined with respect to lat/lon coordinates.

This will be explained in more detail later. Before doing this, it is useful to understand the way the 52°North terrainServer works.

2.2 terrainServer operating principle

Let's assume you query elevation information for a region that is given by its bounding box (which refers to some coordinate reference system). The terrainServer application will transform this bounding box into geographic coordinates (lat/lon) first. After this, the software will determine all tiles that are touched by this bounding box. It is assumed that the corresponding ArcInfo ASCII grid (which refers to some coordinate reference system that must not be geographic) contains elevation information for the complete tile region!

The software then accesses all the relevant elevation grids, performs the interpolation procedure (resampling) and generates the response, which depends on the kind of request.

2.3 Provision of ArcInfo ASCII grid files

Many available GIS and terrain modeling applications offer functionality to convert elevation data into ArcInfo ASCII grid format, e.g. ArcGIS/ArcInfo, FME etc. Note that a simple converter is available as Java source code inside the 52°North Triturus library. Nonetheless, if your elevation data is given in a cell-based structure, experienced programmers should be able to implement such a converter. See appendix B of this document for a description of the ArcInfo ASCII grid format. A proven means to generate ArcInfo ASCII grids is to provide a set of polygons defining the tiling and to clip the original elevation points according to these geometries.

The 52°North terrainServer uses the German TK 25 "Blattschnitt" by default. This can be changed by editing the configuration file $TERRAIN_SERVER_HOME/WEB-INF/classes/tiledef.properties. Note that the terrainServer will locate a tile for a given position based on the position's lat/lon coordinates! If you want to use an individual tile structure, see appendix C.

You must provide the ArcInfo ASCII grids according to the tile structure used for every coordinate reference system that should be supported. An ArcInfo ASCII grid file must be present for every tile defined and every coordinate reference system to be supported. For performance reasons, we recommend that the grid size be neither too big nor too small. We've had good experience with grid sizes in the range of 250x250 to 500x500 cells.

2.4 File organization

Each requested tile requires an ArcInfo ASCII grid file referring to the given coordinate reference system. The file must be named as follows:

<tiling specifier>/epsg_<crs>/dgm<no>.asc

Here, <tiling specifier> is the tiling identifier, <crs> denotes the coordinate reference system's EPSG code, and <no> the corresponding tile number (cf. appendix C).

Note: For the German TK 25-"Blattschnitt", <tiling specifier> is set to "TK25" and <no> gives the "Blattnummer".

The current 52°North terrainServer release supports the coordinate reference systems listed below.

EPSG code	Coordinate reference system
25832	UTM ETRS 1989 Zone 32 North, given in meters.
31466	Zone 2 (2. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point).
31467	Zone 3 (3. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point).
31468	Zone 4 (4. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point).
31492	Zone 2 (2. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point). Note that corresponding elevation data have to be provided in the folder epsg_31466!
31493	Zone 3 (3. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point). Note that corresponding elevation data have to be provided in the folder epsg_31467!
31494	Zone 4 (4. Streifen) Gauß-Krüger, given in meters (7-digit before decimal point). Note that corresponding elevation data have to be provided in the folder epsg_31468!
32632	UTM Zone 32 North, given in meters. Note that corresponding elevation data have to be provided in the folder epsg_25832!

4326	Geographic coordinates (WGS-84), given in decimal degrees.

Example: Your investigation area covers the tiles #8333, #8334, #8433 und #8434 (this might be German "TK 25-Blatt" numbers, or just the numbers of your specific tiles). The figure below illustrates the tiles' locations. Each tile (and for all coordinate reference systems that shall be supported by your terrain services) requires an ArcInfo ASCII grid file.

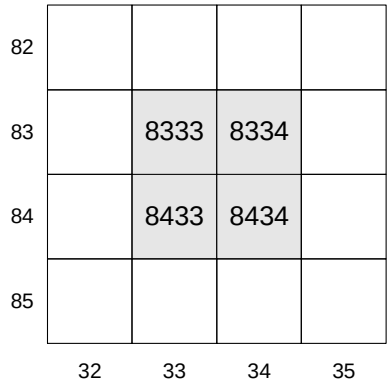


Figure: Spatial location of the tiles #8333, #8334, #8433, and #8434

Assuming your terrain service should support UTM zone 32 North and Gauß-Krüger Zone 3, these 8 grid files must be prepared:

<tiling specifier>/epsg_25832/dgm8333.asc
<tiling specifier>/epsg_25832/dgm8334.asc
<tiling specifier>/epsg_25832/dgm8433.asc
<tiling specifier>/epsg_25832/dgm8434.asc
<tiling specifier>/epsg_31467/dgm8333.asc
<tiling specifier>/epsg_31467/dgm8334.asc
<tiling specifier>/epsg_31467/dgm8433.asc
<tiling specifier>/epsg_31467/dgm8434.asc

2.5 Practical hints for elevation data preparation

Elevation data must be provided for all locations, respectively bounding boxes, which are potentially requested by client applications. Keep in mind the aspects mentioned below:

> *Complete tile coverage:* Each single ArcInfo ASCII grid's elevation values must cover the corresponding tile completely. An elevation value must be present for all points inside a tile. If value interpolation for a terrain service fails, an error message usually appears. E.g., the Web Terrain Servlet would deliver one of the following error messages:
"Missing elevation information (<no>, <crs>)", if the tile is not present, or
"Did not assign values to all grid cells", if tile elevation values are not available for all points inside the tile (often points in the border area of a tile are missing).

> *Consideration of the tile's rotation angle:* ArcInfo ASCII grids do not allow the specification of rotation angles. Since the tiles must not be oriented parallel to the x-axis and y-axis in different coordinate reference systems, the ArcInfo ASCII grids usually

have to be larger than the tile. Values outside the tile might be set to NODATA (cf. appendix B).

> *Overlap areas:* ArcInfo ASCII grids might overlap. Nonetheless, overlap areas should be kept small, otherwise interpolation algorithms might be slowed down.

> *Elevation precision:* With respect to the spatial extent of the regions that are typically queried, the precision of z-values should be higher than necessary. E.g., you can keep your ArcInfo ASCII grid files small in size by using integer valued elevations.

> *Elevation data in border regions:* To ensure error free service operation, it is essential to provide elevation data for the border regions of your area. E.g., in the figure below, no elevation grid can be set up for the bounding box B, since there is a tile without a corresponding ArcInfo ASCII grid file. Hence, data is missing and the interpolation algorithm fails.

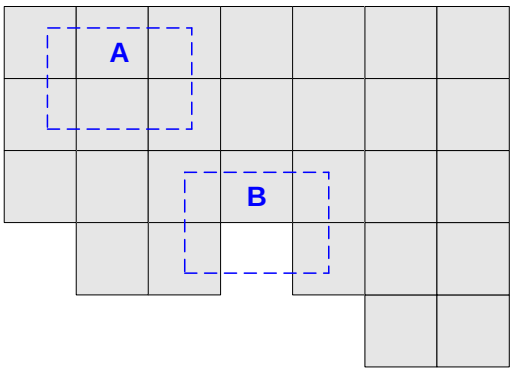


Figure: Tile coverage for some bounding boxes.

3 terrainServer Application Configuration

3.1 Configuring web.xml

The file $TERRAIN_SERVER_HOME/WEB-INF/classes/web.xml is the main configuration file for the whole terrainServer application. The following table shows all configuration attributes and adds some explanatory information. A list of all initialization parameters can be found in appendix A of this document.

Parameter name	Servlets	Short description	Default value
CapabilitiesFile	all	Complete path for the XML file that will be sent back as response to GetCapabilities requests. This parameter should be configured for all services (WTS, W3DS, Cross-section service/ProfileServlet) offered by the terrainServer. Note that the given paths must lead to files that are locally reachable from the Web server.	(none)
SourceGridPath	all	Directory for elevation source data (ArcInfo ASCII grids). See chapter 2 of this document. This parameter must be configured for all services (WTS, W3DS, Cross-section service/ProfileServlet) offered by the terrainServer. Note that the given paths must lead to files that are locally reachable from the Web server.	(none)
PovrayInstallationPath	WebTerrainServlet	Installation path for your POV-Ray rendering engine (needed for WTS support only). See chapter 6 of this document.	(none)
WorkingDirectory	all	Working directory for service operations. This is the location, where temporary files will be placed. This parameter must be configured for all services (WTS, W3DS, Cross-section service/ProfileServlet) offered by the terrainServer. Note that the given paths must lead to files that are locally reachable from the Web server.	(none)
DefaultDrape	WebTerrainServlet	Image file to be draped to a relief if no drape URL is given (used for WTS services only).	(none)

Example:

```
<init-param>
  <param-name>SourceGridPath</param-name>
  <param-value>/52n_terrainServer/demdata</param-value>
</init-param>
```

3.2 Logger configuration

3.2.1 Specific log files for application

Specific request log information is written to temporary text files, which are placed inside the application's working directory; see appendix A.2, description of initialization parameter WorkingDirectory.

The log files will be named according to the following pattern:

~<number>_<id>.log

Here, <number> gives the current request number. In combination with the <id>, a unique name key is given. Note that the log4j-logger logs this file name too (see paragraph 3.2.2).

3.2.2 Log4j logger

You may need to increase the log level of the terrainServer. Therefore, open the file $TERRAIN_SERVER_HOME/WEB-INF/classes/log4j.xml.

Basically, the file contains the following information:

```
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
   <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
      <param name="Threshold" value="WARN"/>
      <layout class="org.apache.log4j.PatternLayout">
         <param name="ConversionPattern" value="[%-5p] %d [%t] %c - %m%n"/>
      </layout>
   </appender>
   <appender name="terrainServerLog" class="org.apache.log4j.RollingFileAppender">
      <param name="File" value="${catalina.base}/logs/terrainServer.log"/>
      <param name="Threshold" value="INFO"/>
      <param name="MaxFileSize" value="5000KB"/>
      <param name="MaxBackupIndex" value="3"/>
      <layout class="org.apache.log4j.PatternLayout">
         <param name="ConversionPattern" value="[%-5p] %d{ISO8601} [%t] %c - %m%n"/>
      </layout>
   </appender>
   <logger name="org.52n.v3d.terrainServer">
      <level value="INFO"/>
   </logger>
```

```
   <logger name="org.52n.v3d.triturus">
      <level value="INFO"/>
   </logger>
   <root>
      <level value="WARN"/>
      <appender-ref ref="CONSOLE"/>
      <appender-ref ref="LOGFILE"/>
   </root>
</log4j:configuration>
```

If necessary, the output location of the log file can be changed. The level of detail of the logging can be determined by setting the level value from INFO to DEBUG.

4 Servlet Container Configuration (Apache Tomcat)

This Apache Tomcat folder will be referenced as %TOMCAT_HOME% in the next chapters.

4.1 Context configuration

Whether you want to run the terainServer within your $TOMCAT_HOME/webapps folder or not, it may be necessary to create a terrainServer.xml file to be put into the $TOMCAT_HOME/conf/Catalina/localhost folder.

The file definition looks like this:

```
<Context
        path="/terrainServer"
        docBase="[$TERRAIN_SERVER_HOME]"
        debug="0"
        reloadable="true">
</Context>
```

Please note that the docBase attribute needs to point to the terrainServer installation folder.

4.2 Starting up the Tomcat server

Start up the Tomcat server from your command line. Change to the folder $TOMCAT_HOME/bin and execute the batch command file named startup. Set the environment variable JAVA_HOME in advance. It should point to your JDK. Assuming you are using JDK 1.6, under UNIX systems you would enter

```
JAVA_HOME=/.../java/jdk1.6
cd $TOMCAT_HOME/bin
./startup
```

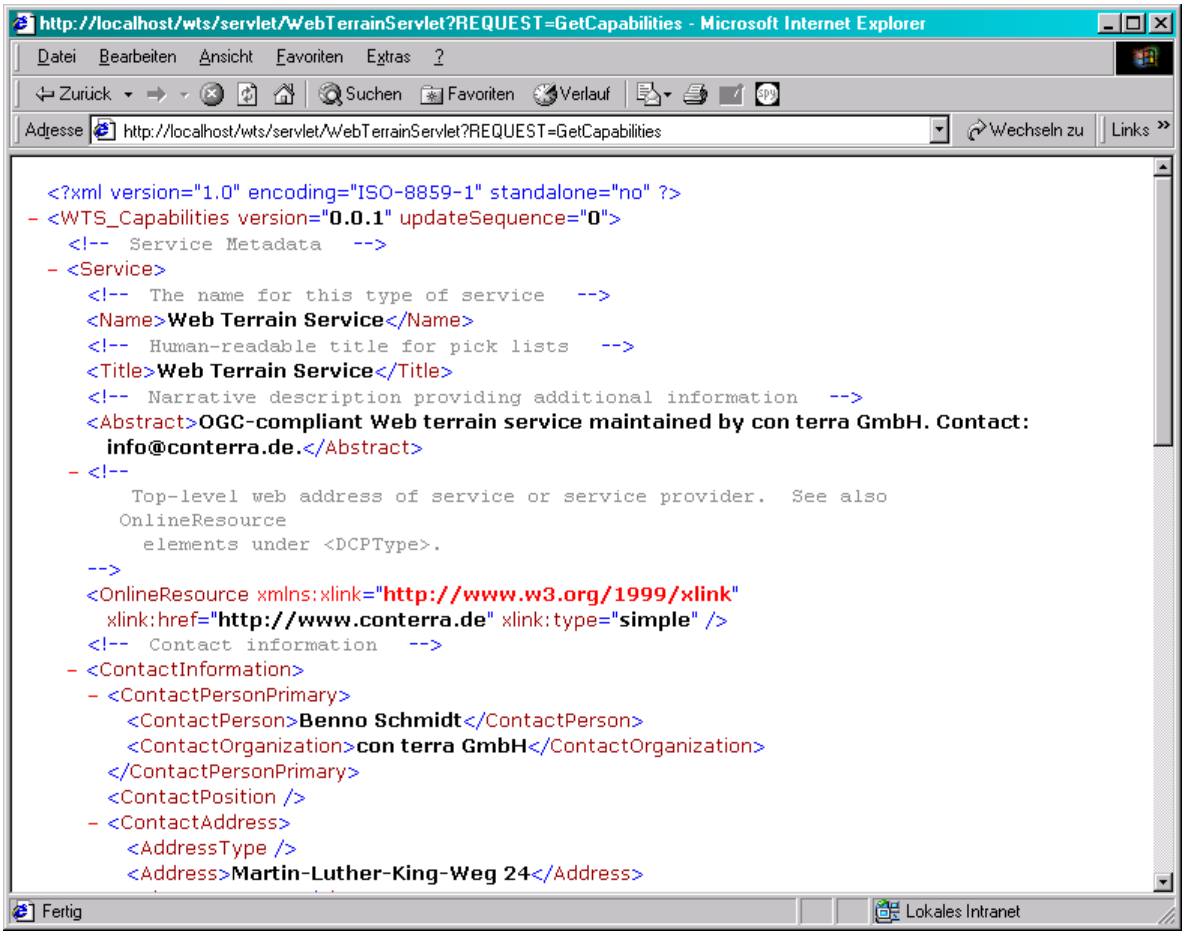
whereas the Windows equivalent would be:

```
JAVA_HOME=...\Java\jdk1.6
cd %TOMCAT_HOME%\bin
startup
```

To restart the server, enter shutdown and then startup again.

5 Test Application

After carrying out the steps described in the chapters above (re)start your Apache Tomcat and open a Browser. E.g., type http://[host]/terrainServer/DEMServlet?REQUEST=GetCapabilities and see the basic terrain service's capabilities. The result should like this:



For more request examples, see the service descriptions of the 52°North terrainServer.

Enjoy!

6 POV-Ray Installation and Configuration

If you want to set up the 52°North terrainServer's Web Terrain Service (WTS), you must provide a suitable rendering engine. The 52°North terrainServer currently supports the usage of the POV-Ray engine (short for "Persistence of Vision Raytracer") to perform server side rendering and generate perspective images.

If you do not intend to set up a WTS, you do not have to install the POV-Ray rendering engine!

You can get the POV-Ray software via http://www.povray.org (open and free of charge). Please pay attention to the specific POV-Ray terms of use!

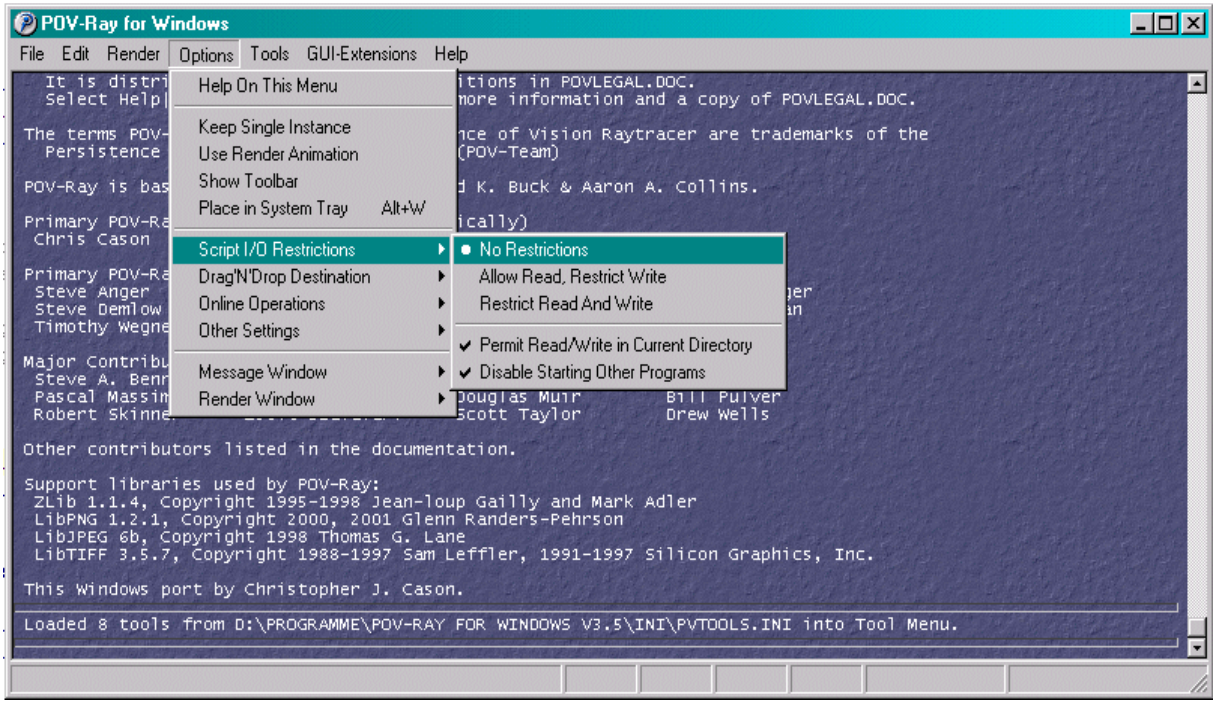
The POV-Ray engine version 3.6 is recommended, however, version 3.5 should be suitable as well.

To install POV-Ray under Windows operation systems, choose the command "Software" → "Add software" in your control panel. Important note: Avoid executing the file povwin36.exe (resp. povwin35.exe for version 3.5) directly.

Follow the instructions given by the POV-Ray installation routine. Note that the options "Backup Replaced Files", "Automatically Check For Updates" and "Include System Information" must not be activated.

For installations on Linux systems, please pay attention to the specific hints for Linux given by the POV-Ray distributors.

The figure below shows the POV-Ray user interface to configure the rendering engine under Windows.



Once the installation process is complete, the rendering engine must be configured according to the 52°North terrainServer's demands. Please follow these steps:

> Start "POV-Ray for Windows" (e.g., using the Windows "Start" button).
> Allow simultaneous execution of more than a single POV-Ray instance! This can be done with the settings listed below, found inside POV-Ray's "Options" menu:
  - "Keep Single Instance" must be deactivated (click this option, so that no ✓ will be shown)
  - "Use Render Animation" must be deactivated (mostly, this is the default value)
  - "Show Toolbar" must be deactivated.
> Under "Options" → "Script I/O Restrictions" the following settings are necessary:
  - "● No Restrictions must be active" (mostly, this is the default value)
  - "✓Permit Read/Write in Current Directory" must be active (mostly, this is the default value)
> Under "Options" → "Other Settings" these settings are necessary:
  - "Use Editor" deactivated (note that POV-Ray will restart automatically)
  - "Show Tips of the Day" deactivated
  - "Show Tooltips" deactivated
  - "Preserve Messages" deactivated
  - "✓ Save Settings On Exit" must be active (mostly, this is the default value)
  - "Hide New User Help" deactivated (mostly, this is the default value)
> Under "Options" → "Render Window" these settings are required:
  - "✓ Force 8-bit Display" active
  - "✓ AutoClose" active
  - "Keep Above Main" deactivated
  - "Get Focus On Creation" deactivated
  - "✓ Hide When Main Minimized" active (mostly, this is the default value)
> Under "Render" → "On Completion..." choose the setting "Exit POV-Ray for Windows" now.
> Call "Tools" → "Edit PVENGINE.INI". A text editor will pop up. Now change the ItsAboutTime value to -1. Then save the text file and close the text-editor.
> Close and restart "POV-Ray for Windows".
> Deactivate the switch "Options" → "Other Settings" → "Save Settings On Exit".
> Close "POV-Ray for Windows".

Now POV-Ray should be configured as needed by the terrainServer's WTS Servlet.

Note: For further information, refer to POV-Ray's online help.

Appendix A: Web.xml initialization parameters

A.1 Overview

The table below gives a brief overview over the application's initialization parameters. More detailed parameter descriptions will be given subsequently.

Parameter name	Servlets	Short description	Data type / Default value
CapabilitiesFile	all	Complete path for the XML file that will be sent back as response to GetCapabilities requests.	String (file path) / none
SourceGridPath	all	Directory for elevation source data (ArcInfo ASCII grids). See paragraph 3.1 of this document.	String (file path) / none
TileLocator	all	Tile definition file, see chapter 2 resp. appendix C.	String / "TK25"
MinCellSize	all	Minimum cell size of elevation models for requests using metric coordinate reference systems.	Floating point number > 0 / 50 (meters)
MinCellSizeLatLon	all	Minimum cell size of elevation models for requests using geographic coordinates (lat/lon).	Floating point number > 0
SearchRadiusMin	WebTerrain-Servlet, DEMServlet	Minimum search radius for gridding operation (elevation value resampling).	Floating point number > 0 / 49.99
DefaultDrape	WebTerrain-Servlet	Image file to be draped to a relief if no drape UTRL will be specified (used for WTS services only).	String
PovrayInstallation-Path	WebTerrain-Servlet	Installation path for your POV-Ray rendering engine (needed for WTS support only).	String (file path)
PovrayExec	WebTerrain-Servlet	Name of POV-Ray executable.	String (file path)
ShellCommand	WebTerrain-Servlet	Shell command parameters.	String

Parameter name	Servlets	Short description	Data type / Default value
ShellCommandQuot	WebTerrain-Servlet	Switch, which enables the support for blank characters to specify `PovrayInstallationPath` and `PovrayExec` under Windows.	Boolean
WorkingDirectory	all	Working directory for service operations. This is the location, where temporary files will be placed on the server.	String
WebConnectProxySet	WebTerrain-Servlet	Flag determining whether a proxy server is used to establish Web connections (relevant for WMS and drape image access). Hint: Better avoid using this switch.	Boolean
WebConnectProxy-Host	WebTerrain-Servlet	Proxy server name, if WebConnect-ProxySet is set.	String
WebConnectProxy-Port	WebTerrain-Servlet	Port number for proxy connections, if Web-ConnectProxySet is set.	Integer number > 0
WebConnectNon-ProxyHosts	WebTerrain-Servlet	Name list of non proxy servers (if Web-ConnectProxySet is set)	String list (|-symbol as separator)
CopyrightText-Content	WebTerrain-Servlet	Content of the copyright text that will be rendered at the lower left corner of WTS images.	String
CopyrightTextFont	WebTerrain-Servlet	Font name to be used for copyright text rendering.	String
CopyrightTextSize	WebTerrain-Servlet	Font size to be used for copyright text rendering.	Integer number
CopyrightText-Color	WebTerrain-Servlet	Color value to be used for copyright text rendering and for the north arrow symbol.	String holding hexadecimal-coded RGB-values (format `0xRRGGBB`)
PedestalColor	WebTerrain-Servlet	Color to be used to render a relief pedestal.	String holding hexadecimal coded RGB-values (format `0xRRGGBB`)

Parameter name	Servlets	Short description	Data type / Default value
TimeSliceDuration	WebTerrain-Servlet	Time slice length given in milliseconds (if Use-TimeSlices is set).	Integer number > 0 / 4000
RendererTimeout	WebTerrain-Servlet	Threshold specifying the time span that has to pass, before unfinished rendering attempts will be regarded as unsuccessful.	Integer number > 0 (msec) / 20000
RendererImmediate-Termination	WebTerrain-Servlet	Internal switch to deactivate waitFor mode (RedHat-Linux).	Boolean (default: `false`)
MonProtUser	WebTerrain-Servlet	User name used to monitoring protocol requests (currently limited to a single user).	String
MonProtPasswd	WebTerrain-Servlet	Password for the user specified in Mon-ProtUser.	String
MaxArea	all	Maximum size of the queryable bounding box, usually given in m^2.	Floating point number > 0
ErrMsgFile	WebTerrain-Servlet	Optional parameter specifying a file to define error messages freely.	String (file path)

Note: The web.xml is placed in the application's WEB-INF folder. For more configurable parameters, investigate this file with its comments and the underlying Java source code.

The initialization parameter's values will be read every time you start your Servlet engine (e.g. Tomcat). Thus, file edits will not be considered while running the terrainServer. If necessary, the Web application must be explicitly restarted.

A.2 Detailed parameter descriptions

Some of the initialization parameters will now be explained in more detail.

CapabilitiesFile (mandatory)
CapabilitiesFile points to the XML document that will be returned to the calling client as response to GetCapabilities requests. This path usually has to be adapted to your individual environment. Note that this parameter has to be set for all Servlets (WebTerrainServlet, ProfileServlet, DEMServlet) you wish to deploy separately.

The terrainServer Web application download contains a sample XML file. Here are some important document entries:

- Service/Abstract: Put in a short description of your service here.
- Service/OnlineResource, Capability/Request/GetCapabilities/DCPType/HTTP/Get/Online-Resource and Capability/Request/Get.../DCPType/HTTP/Get/OnlineResource: Put in the URLs for your deployed service. E.g., for a Web Terrain Service Capability/Request/GetView/DCPType/HTTP/Get/OnlineResource should point to the URL performing a WTS-GetView-request.
- Service/ContactInformation: Name a contact person inside your organization here.

SourceGridPath (mandatory)
SourceGridPath points to the folder in which your ArcInfo ASCII grid files with the elevation data sources can be found. See chapter 2. Note that this folder should **not** be part of the folder structure that is deployed as a Web application!

The parameter SourceGridPath has to be set for all Servlets (WebTerrainServlet, ProfileServlet, DEMServlet) you wish to deploy separately.

PovrayInstallationPath (mandatory for WebTerrainServlet)
This specifies the path where POV-Ray has been installed. See chapter 6 for a detailed description of the POV-Ray installation process.

If the path name consists of blank characters, some Windows systems errors might occur when instantiating the POV-Ray rendering engine. In this case, we recommend enclosing the folder name with \" characters, e.g. C:\Programs\"POV-Ray 3.6 for Windows\", or (better!) avoiding blank characters inside folder names.

PovrayExec (mandatory for WebTerrainServlet)
Under Windows, the executable POV-Ray rendering engine pvengine.exe is usually located inside the sub-folder bin. For UNIX installations, the pvengine is often located in the POV-Ray root folder. Check your personal POV-Ray installation!

PovrayWin (mandatory for WebTerrainServlet)
For Windows installations, this switch is usually set to true, while for UNIX installations, the value false is preferred. If this parameter is missing, the default value true (Windows) will be set.

ShellCommand (mandatory for WebTerrainServlet)
ShellCommand determines the command that will be called by the underlying operating system to start a shell for POV-Ray execution. For Linux systems, no value must be given (i.e., an empty string), since the rendering engine can be invoked directly.

For security reasons, it is recommended not to use the value cmd.exe. For example, one might remove the cmd.exe from the Windows installation folder and use a renamed cmd.exe file (e.g. named \mySpecialFolder \povraystarter.exe) instead.

ShellCommandParams (optional)
Here, parameters for the command that has been specified via ShellCommand, can be given. E.g., under Windows operating systems, `/E:1900 /C` to invoke a shell via `cmd.exe /E:1900 /C`

This parameter is usually not needed for Linux installations.

ShellCommandQuot (optional)
If this switch is set to true, blank spaces (" ") are allowed to specify PovrayInstallationPath and PovrayExec under Windows. The path resulting from the specified folder and the executable name will be set in quotations ("). Nonetheless, we recommend **not** to use blank spaces in folder names.

WorkingDirectory (mandatory)
This parameter specifies the server side working directory. The application will place the input files required by the POV-Ray rendering engine (scene description, data files, etc.), as well as other temporary files in this directory. Moreover, you will also find specific application log files that have been generated during Servlet operations; see paragraph 3.2.1 of this guide.

DefaultDrape (optional)
This parameter defines the drape file to be used if there is no value present for the DRAPE parameter inside the Web Terrain Service request. Usually, the path given for DefaultDrape has to be adapted for the web.xml that comes with the "52°North terrainServer".

MaxArea (mandatory)
MaxArea specifies the maximum size of the queryable extent. The value has to be given in square meters. Before changing the preset value, make sure that your system performance allows for extent calculations with the given size in a reasonable time. (CPU performance is a significant factor.)

MinCellsize (optional)
MinCellsize specifies the minimum elevation grid cell size that will be used to construct elevation grids for WTS services. Hence, this value might influence the visual impression (spatial resolution) of the resulting perspective views.

For DEM and W3DS services, MinCellsize gives the minimum value that might be queried (CELLSIZE parameter in HTTP Get requests). This parameter is used for metric coordinate reference systems. The value must be given in meters.

Usually, the value to be set depends on your elevation data characteristics and on the finest resolution you want to offer via the terrainServer. By default, the value is set to 50.

MinCellsizeLatLon (optional)
Corresponding to MinCellsize, the value of the initialization parameter MinCellSizeLatLon determines the minimum cell size if geographic coordinates will be used, e.g. queries with SRS="EPSG:4326". The value must be given as a floating point number in decimal degrees. The default value is 4,629627·10-4 (to be specified as 4.629627E-4 inside the web.xml).

SearchRadiusMin (optional)
SearchRadiusMin gives the minimum search radius for the grid resampling process that will be performed inside the terrainServer Servlet. The value hast to be given in meters (both for metric coordinate reference systems and queries referring to geographic coordinates).

ErrMsgFile (optional)
By default, this parameter is not set (empty string). You have to specify this parameter if you want to change the text of error messages that might be returned by a service.

Optionally, you might want to specify an ASCII file (incl. path) giving the error messages. This ASCII file holds the internal error numbers and the message text line by line in a comma separated format. For the text strings, a |-symbol represents a carriage return/line feed. A sample file comes with your terrainServer WAR file.

Appendix B: ArcInfo ASCII grid format

The 52°North terrainServer assumes elevation data to be present as ArcInfo ASCII grids files. In fact, this format is kept very simple and suitable to hold elevation data referring to regular cellular grid geometries. Note that these grids must be parallel to x-axis respectively to y-axis, and that the cell sizes both in x and y directions must be equivalent for all grid cells. Elevation values refer to the center point of the cells. The origin gives the lower left corner of the underlying grid geometry. I.e., the origin differs from the cell center by half a cell width in each direction. ArcInfo ASCII grid files are structured as follows:

```
ncols <ncols>                       /* Number of columns */
nrows <nrows>                       /* Number of rows */
xllcorner <x>                       /* x-coordinate of lower left corner */
yllcorner <y>                       /* y-coordinate of lower left corner */
cellsize <size>                     /* cell-size both for x- and y-direction */
NODATA_value <NODATA>               /* Value used for unset grid cells */
<z_1,1> <z_1,2> <z_1,3> ... <z_1,ncols>      /* Elevation values for first row */
<z_2,1> <z_2,2> <z_2,3> ... <z_2,ncols>      /* Elevation values for second row */
...
<z_nrows,1> <z_nrows,2> <z_nrows,3> ... <z_nrows,ncols> /* Elevation values for last row */
```

A simple sample file could look as follows:

```
ncols           10
nrows           8
xllcorner       3522415
yllcorner       5815405
cellsize        50
NODATA_value    -9999
124 128 128 128 126 125 124 124 123 122
125 126.5 127.5 127 126 -9999 124 125 124 123
127 -9999 127 128 126 -9999 125 124 124 123
128 128 128 128 126 125 124 124 123 122
129 124 127 128 128 -9999 128 124 123 124
128 -9999 128 127.5 128 -9999 128 124 123 125
124 124.5 128 128 126 125 124 124 122.5 124
123 124 127 128 128 -9999 128 124 123 122
```

Appendix C: Tile number calculation formulary

This appendix gives the formulary that is used by the 52°North terrainServer to determine tile numbers from geographic coordinates.

C.1 Linear mapping of four digit tile numbers

Parameters:

λ, φ	Geographical longitude and latitude in degrees
a_i	Tile mapping parameters

Computation formula:

b_{12}, b_{34}	Tile number parts (as real numbers, $0 \leq b_{ij} < 100$)
B_1, B_2, B_3, B_4	Tile number as four digit character string

$$\begin{pmatrix} b_{12} \\ b_{34} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_4 & a_5 \end{pmatrix} \cdot \begin{pmatrix} \lambda \\ \varphi \end{pmatrix} + \begin{pmatrix} a_3 \\ a_6 \end{pmatrix}$$

B_iB_j results from $\underline{b}_{ij}$ (smallest number that rounds up to b_{ij}), where for single digit values a leading '0' will be put in front, i.e. B_iB_j = "12" for b_{ij} = 12,5 respectively B_iB_j = "08" for b_{ij} = 8.

C.2 German TK 25 tiles ("Blattnummern")

Here, these parameters will be used:

$$A = \begin{pmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \end{pmatrix} = \begin{pmatrix} 0 & -10 & 560 \\ 6 & 0 & -34 \end{pmatrix}$$

Note: This tiling covers the area of Germany, as well as the eastward region (extending to East Prussia/Silesia) and parts of the des Alpine region (5 2/3° to 22 1/3° East, 46° to 56° North).

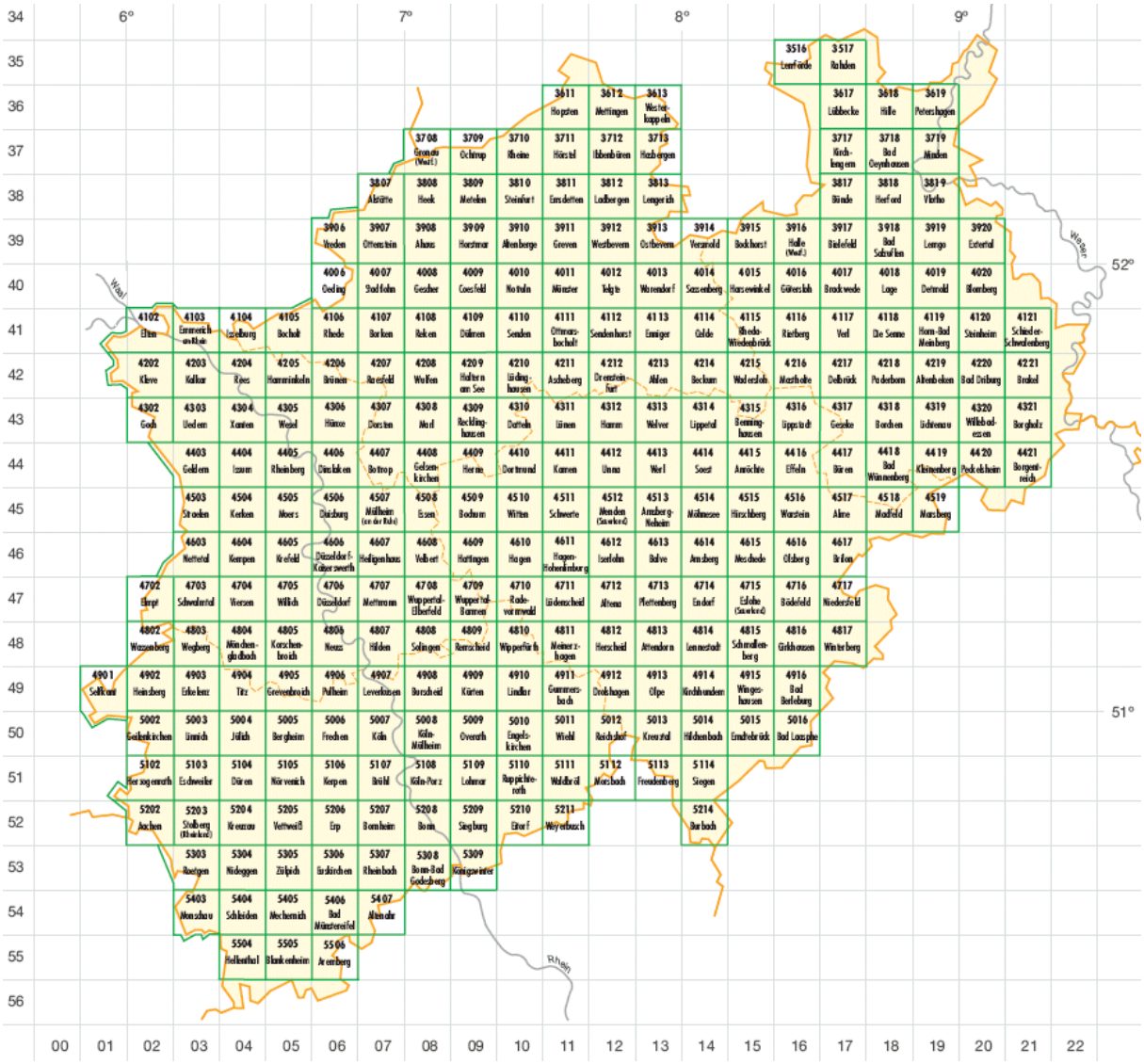


Figure: TK 25-Blattschnitt of Northrhine-Westphalia, Germany (Source: Geobasis NRW, Köln)

Example: For $\lambda = 7^\circ 36'$, $\varphi = 51^\circ 40'$, the result will be $\underline{b}_{12} = 43$, $\underline{b}_{34} = 8$ and thus the tile number (Blattnummer) will be $B_1B_2B_3B_4$ = "4311".

C.3 Alternative tile definitions

A few examples of some alternative tiling definitions are as follows.

Here are the parameters for a definition covering the whole Earth. The tiles used cover regions of 10° x 10° degrees (tile numbers "0000" for S90 W180 to "3517" for N90 O180):

$$A = \begin{pmatrix} 0{,}1 & 0 & 18 \\ 0 & 0{,}1 & 9 \end{pmatrix}$$

The parameter for a worldwide definition of 5° x 5° sized tiles would look like this ("0000" for S90 W180 to "7135" for N90 O180):

$$A = \begin{pmatrix} 0{,}2 & 0 & 36 \\ 0 & 0{,}2 & 18 \end{pmatrix}$$

Here are the parameters for a definition that will be usable Europe-wide, where each tile covers a region of 1° x 1° ("0000" for N30 W30 to "9959" for N90 O70):

$$A = \begin{pmatrix} 1 & 0 & 30 \\ 0 & 1 & -30 \end{pmatrix}$$

Parameters for a tiling that covers the main part of Europe (without the Canary Islands, eastwards to Moscow and a bit beyond) with tiles of 0,5° x 0,5° in size ("0000" for N31 W11 to "9999" for N81 O39):

$$A = \begin{pmatrix} 2 & 0 & 22 \\ 0 & 2 & -62 \end{pmatrix}$$

C.4 Computation of tile positions from geographic coordinates

To compute a tile's bounding box from lat/lon coordinates, just invert the given formula:

$$\begin{pmatrix} \lambda_{\min} \\ \varphi_{\min} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_4 & a_5 \end{pmatrix}^{-1} \cdot \begin{pmatrix} \underline{b}_{12} - a_3 \\ \underline{b}_{34} - a_6 \end{pmatrix} \qquad \begin{pmatrix} \lambda_{\max} \\ \varphi_{\max} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_4 & a_5 \end{pmatrix}^{-1} \cdot \begin{pmatrix} \overline{b}_{12} - a_3 \\ \overline{b}_{34} - a_6 \end{pmatrix}$$

Note: An underlined number b denotes the biggest (integer) number that is < b, an overscored number, the smallest (integer) number > b.

Appendix D: Frequently asked questions (FAQs)

D.1 Can the services process elevation data worldwide?

On principle, yes!

Due to the tiling mechanism used in older terrainServer releases, there has been support for Germany, Poland, Switzerland, the Czech republic, Luxembourg, and Austria only (longitude 5 2/3° to 22 1/3° East, latitude 46° to 56° North). 52°North terrainServer allows the free definition of tiling structures.

Please consider that not all coordinate reference systems (CRS) used worldwide are supported yet! However, "EPSG:4326" (geographic coordinates) should work. Additional CRS support has to be implemented (see Java source code) to support local coordinate reference systems that do not appear in the table given in paragraph 2.4 of this guide.

D.2 Which data formats will be accepted for digital elevation data?

The terrainServer assumes you are using ArcInfo ASCII grids. These grids have to be organized in a suitable tile structure. See chapter 2 of this guide for details.

D.3 Is it possible to connect the 52°North terrainServer to 2D mapping clients?

Yes, this is possible and highly desirable! There are a lot of interesting use cases that require the linkage of terrain services and 2d map clients. E.g.: Display a WMS image in the map viewer, push a "3D button", and display the WMS image as perspective view generated by the 52°North terrainServer. A suitable application to support interactive viewing of the terrainServer generated WTS images, is the 52°North terrainViewer. An example for a 2d Web mapping application integrating the 52°North terrainViewer, is the sdi.suite mapClient 2 (available via con terra GmbH, Germany).

D.4 Does the 52 terrainServer support my operation system?

Currently, the software should run well on Windows operation systems (XP, 200x, etc.) and the Linux versions RedHat AS4.0 und SuSe 9.3.

D.5 Do I need special 3D graphics hardware?

No.

D.6 How can X11 errors be resolved?

Under UNIX operation systems, the Servlets might respond with X11 errors to your requests, e.g. "can't connect to X11 window server using ':0.0'...". Under Linux, the Servlet engine (Tomcat) should be started using the JVM option java.awt.headless. E.g., set the environment variable JAVAOPTS before starting your Tomcat engine:

```
export JAVAOPTS=-Djava.awt.headless=true
```

D.7 How to resolve OutOfMemoryErrors?

If any memory errors occur (e.g. a Java stack trace showing a java.lang.OutOfMemoryError in your log file), you should make sure enough free memory is available. Check the amount of

free memory on the machine (respectively the hard disk partition) where the Servlet container (e.g., Tomcat) is running. Often, the JVM configuration should be changed, e.g. heap space is running out. You can use the switches -ms und -mx to specify initial resp. maximum heap memory size for your JVM. E.g., this configuration assigns 256 MB of heap memory:

-Xms256m –Xmx256m

Note for Tomcat users: To set the JVM options at Tomcat start time, you can use the environment variable JAVA_OPTS (or CATALINA_OPTS resp. TOMCAT_OPTS). If Tomcat has been installed as a Windows service, corresponding settings have to be done in the Windows registry. See Tomcat documentation.