

Copyright Notice

Copyright 2003 *lat/lon*

The companies and organizations listed above have granted the Open GIS Consortium, Inc. (OGC) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version.

This document does not represent a commitment to implement any portion of this specification in any company's products.

OGC's Legal, IPR and Copyright Statements are found at <http://www.opengis.org/legal/ipr.htm>.

Permission to use, copy, and distribute this document in any medium for any purpose and without fee or royalty is hereby granted, provided that you include the above list of copyright holders and the entire text of this NOTICE.

We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

No right to create modifications or derivatives of OGC documents is granted pursuant to this license. However, if additional requirements (as documented in the Copyright FAQ at http://www.opengis.org/legal/ipr_faq.htm) are satisfied, the right to create modifications or derivatives is sometimes granted by the OGC to individuals complying with those requirements.

THIS DOCUMENT IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with copyright holders.

RESTRICTED RIGHTS LEGEND. Use, duplication, or disclosure by government is subject to restrictions as set forth in subdivision (c)(1)(ii) of the Right in Technical Data and Computer Software Clause at DFARS 252.227.7013

OpenGIS® is a trademark or registered trademark of Open GIS Consortium, Inc. in the United States and in other countries.

Note: This document is not an OGC Standard. Internal and external documents cannot refer to it as such. Drafts are distributed for review and comment and are subject to change without notice.

deegree Web Map Server

deegree is a Java Framework product for the implementation of local and web based GIS applications. Its interfaces and architecture guarantee optimized interoperability due to the compliance with OpenGIS standards. **deegree** is available as an open source project at www.deegree.org (under the terms of the GNU Lesser General Public License since August 2002).

Besides a WMS, **deegree** comprises a WCS and a WFS services.

deegree's WMS server is flexible concerning the possibilities of configuration, the adaptation to different data sources and formats, layouts and server environments. The configuration of the WMS, the Catalog Service and the WFS takes place by customizing different XML files that control the functionality of the **deegree** server.

The web services of **deegree** are realized as Java servlets, in case of the WMS, there is one general WMS servlet and a special servlet for legend graphic. These servlets have to be integrated in the respective web server. Most of the common web servers support servlet technology, so that **deegree** is not limited to special products. The Apache-Tomcat (4.x) Servlet-Engine is recommended due to the widespread of installations and the status as open-source product.

NOTE: Additional contribution on using **deegree** comes from feedback from Professor Stephan Winter and his students from his class on GI Interoperability. See the last section of Recipe 2 for this very useful feedback.

Step 1: Preliminary Requirements

The following preliminary requirements are needed to run the **deegree** WMS:

- Download **deegree** WMS package from **deegree** WMS page¹ appropriate for your platform (LINUX or Windows) and archive extraction. See **deegree's** online documentation² (Section 2.1) for instructions.
- Tomcat integration. Download and install a current release of Tomcat. See **deegree's** online documentation (Section 2.2) for Tomcat deployment with **deegree** WMS.

Step 2: Installation of deegree WMS

- WMS Configuration

During the initializing process, the name and position of the central configuration file (configuration.xml) will be transferred. The data sources will be registered and some other configuration files will be referenced directly or indirectly. One of these is the WMS capabilities document (WMS capabilities.dtd), that describes the abilities of the WMS. For a detailed description of the structure of the WMS capabilities document, refer to the WMS Specification.

Table 1, below, lists required configuration files and their associated DTD files supplied with **deegree** WMS. See Appendix A for full definition of these files, and examples.

¹ deegree WMS page: <http://deegree.sourceforge.net>

² http://www.deegree.org/demo/WMS-Konfiguration_english.htm

configuration file	description
configuration.xml	Central configuration file for the WMS
configuration.dtd	dtd, that describes the central configuration file
jld.dtd	dtd, that describes the structure of the deegree Layer Descriptors
deegree_styles.xml	Determines the layout of rendering vector data
degConfig.xml	Configures the Display Element Generator Service. References the deegree_styles.xml.
gcdescriptor.dtd	dtd, that describes the structure of the configuration file for the integration of grid data
gridcover.xml	Describes the metadata for the integration of the images in the WMS.
deegree.xml	capabilities-document; describes the abilities of the WMS
feature_info_to_html.xsl	xslt-script for formatting the results of an FeatureInfoRequests

Table 1: deegree WMS Required Configuration Files and Associated DTDs

Note: In the case of using a database as a geospatial data source, a configuration file is required for describing the database linkage.

In case you did not change (as part of the preliminary requirement step - Tomcat integration) the standard settings of the supplied XML files, you now must adjust the directory settings in configuration.xml and then replace the server names in capabilities.xml with the name or IP number of your server, and then you are finished. Otherwise you have to edit degConfig.xml and gridcover_world.xml in the xml directory as well.

Note: In the configuration files supplied with the current version of **deegree** WMS paths, you often have the form:

`http://localhost:8080/deegree/pathname/filename`

This is an optional way to reference configuration files with URLs. If you are not familiar or comfortable with referencing files in this way, stay with the standard (absolute) syntax supplied in the **deegree's** online documentation.

1.1.1.1. deegree's Internal Service Structure

According to A. Cuthbert's portrayal model, a WMS is divided into four services. The filter service arranges the communication between the data sources and supplies a standard interface providing Features. The Display Element Generator (DEG-Service) takes these and combines them with symbolization instructions and supplies display elements. The render service compiles a map from the display elements using meta-information describing the layout. The map will be visualized by the display service.

The display service is realized by a web browser. The other services will be registered by the WMS with one or several java classes. Particularly in case of using several data sources, the WMS will register one java class for each data source. The structure of the central configuration file will be described in the configuration.dtd.

1.1.1.2. DEG- and Render Service(s) Registration

The configuration.dtd describes the architecture of the central **deegree** WMS configuration file.

`<!ELEMENT deegreeConfiguration`

```
(Capabilities,FilterService*,DEGService*,RenderService?)>
```

Each <DEGService> tag and each <RenderService> tag will get a <Class> tag, that knows the name of the java class, which supports a service. The name of the class is stored by the attribute name of the <Class> tag.

```
<!ELEMENT DEGService (class)>
<!ELEMENT RenderService (class)>
<!ELEMENT Class (Config?,Layer*)>
<!ATTLIST Class name CDATA #REQUIRED>
```

The DEG service uses the <Config> tag that is part of the <Class>. The <Config> tag refers to a configuration file that embodies which layout rules will be realized by the DEG service.

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<DEGConfiguration>
  <StyleDocument>
    $deegree_home/xml/deegree_styles.xml
  </StyleDocument>
</DEGConfiguration>
```

1.1.1.3. Configuration File for a DEG Service

The <Layer> tag is not important for the DEG service and the render service at this time. The registration of both services in the WMS could be like the following:

```
<DEGService>
  <Class name="org.deegree_impl.services.wms.degservice.DisplayElementGenerator_Impl">
    <Config>file:/// $deegree_home/xml/degConfig.xml</Config>
  </Class>
</DEGService>
<RenderService>
  <Class
    name="org.deegree_impl.services.wms.renderservice.Renderer_Impl"/
  >
</RenderService>
```

The position of the deg-configuration file must be given in the form of an URL address. The java archive degree.jar contains both of the referenced classes. It is possible to register self-defined classes as services. For this purpose, they have to serve the defined logs and interfaces in **deegree**.

1.1.1.3.1. Testing the Basic Configuration

To test if the basic configuration was done correctly, re-start Tomcat. At this point, the WMS should already be integrated. The following request should return a map of the world (it is assumed that Tomcat uses port 8080):

```
http://localhost:8080/deegree/deegreewms?WMTVER=1.0.0&
REQUEST=map&SRS=EPSG%3A4326&BBOX=-180,-90,180,90&
WIDTH=641&HEIGHT=481&
LAYERS=world,WorldBorder,WorldCities&
STYLES=default,default,citystyle&FORMAT=jpg&
BGCOLOR=0xffff8ff&TRANSPARENT=FALSE&EXCEPTIONS=INIMAGE
```

If you zoom into the map using the following request, you will see that now the cities are visible. This is because the city-layer is defined (see configuration.xml) as valid only for small scales.

```
http://localhost:8080/deegree/deegreewms?WMTVER=1.0.0&
REQUEST=map&SRS=EPSG%3A4326&BBOX=4,47,13,58&WIDTH=641&
HEIGHT=481&LAYERS=world,WorldBorder,WorldCities&
STYLES=default,default,citystyle&FORMAT=jpg&
BGCOLOR=0xffff8ff&TRANSPARENT=FALSE&EXCEPTIONS=INIMAGE
```

1.1.1.3.2. Add your own data to the WMS

All geospatial data to be visualized by **deegree** WMS have to be registered in the capabilities.xml and the main configuration file (configuration.xml). While the first one is the capabilities document described by the OGC WMS Specifications, the later one is a **deegree** WMS specific file.

1.1.1.3.3. Registration of the Filter Service and Data Sources

For the registration of data sources within the configuration.xml file, information about the corresponding filter service is necessary. The <Filterservice> tag contains 0..n <FeatureInfoFormat> tags.

```
<!ELEMENT Filterservice (FeatureInfoFormat*,Class+)>
```

The <FeatureInfoFormat> tag provides information about the XSLT script that is responsible for the formats of the results of a FeatureInfoRequest. Different XSLT scripts for different return formats can be registered. Default is the MIME (HTML) format.

1.1.1.4. Registration of Image Data as Data Source

The **deegree** WMS provides support for images in the following formats: TIFF-, GIF-, BMP-, JPEG- or PNG-image format. For this intention, the class:

```
org.deegree_impl.services.wms.filterservice.GridCoverageFilter
```

must be registered by the <Class> tag in the <Filterservice> tags of the central configuration file (configuration.xml). The **deegree.jar** archive contains that class. Further on, the position of the configuration file with the detailed information of the image data that should be displayed, is described in the <Config> tag. Finally, the WMS gets information about the names of the layer and the assignment of data source and layer.

```
<!ELEMENT Class (Config?,Layer*)>
<!ELEMENT Layer (KnownDataSource+)>
<!ATTLIST Layer name CDATA #REQUIRED>
<!ELEMENT KnownDataSource (#PCDATA)>
<!ATTLIST KnownDataSource minscale CDATA #IMPLIED>
<!ATTLIST KnownDataSource maxscale CDATA #IMPLIED>
```

The referenced data sources (<KnownDataSource>) are specified by the <Config> tag in the XML file. Several data sources can be declared for one layer. Scale dependent visibility may be defined by the attributes minscale and maxscale. This allows for the usage of different image data sources with an optimized resolution for different zoom levels. The complete process of referencing the three layers that have access to image data, could be similar to the following:

```
<Class
name="org.deegree_impl.services.wms.filterservice.GridCoverageFil
ter">
```

```

<Config>file:/// $deegree_home/xml/gridcover.xml</Config>
  <Layer name="sa_hs">
    <KnownDataSource>SachsenAnhalt</KnownDataSource>
  </Layer>
  <Layer name="geomis">
    <KnownDataSource>geomis</KnownDataSource>
  </Layer>
  <Layer name="euro">
    <KnownDataSource>euro</KnownDataSource>
  </Layer>
</Class>

```

In the WMS-Request, the layer will be addressed by "sa_hs", "geomis" and "euro". The images that are assigned to the layer are referenced in the file gridcover.xml. The structure of gridcover.xml is described by gcdescriptor.dtd.

The idea of the configuration file for the implementation of images is the definition of layers. Their names will be referenced in the central configuration file (see above).

```

<!ELEMENT GCLayer (Preview?, ScaledTileCollection+, Property+) >
<!ATTLIST GCLayer name CDATA #REQUIRED>
<!ATTLIST GCLayer minx CDATA #REQUIRED>
<!ATTLIST GCLayer miny CDATA #REQUIRED>
<!ATTLIST GCLayer maxx CDATA #REQUIRED>
<!ATTLIST GCLayer maxy CDATA #REQUIRED>
<!ATTLIST GCLayer cs CDATA #REQUIRED>

```

The further attributes of the <Layer> tag determine the bounding box and the projection parameters. Each layer contains one or more properties for further descriptions. In the case of declaring only one property, a unique Id of the layer is necessary.

```

<!ELEMENT Property EMPTY>
<!ATTLIST Property name CDATA #REQUIRED>
<!ATTLIST Property value CDATA #REQUIRED>

```

Another important aspect is the possibility to define several so called "ScaledTileCollections" within a layer. The ScaledTileCollections have information about the association of image files and layers. The assignment can depend on the zoom level. For each zoom level, a <ScaledTileCollection> will be created. minscale and maxscale correspond to the length of the diagonal of one pixel from the image center, expressed in units of the used spatial reference system. Similar to the layer, a ScaledTileCollection contains properties for further descriptions.

```

<!ELEMENT ScaledTileCollection (Tile+, Property+)>
<!ATTLIST ScaledTileCollection minscale CDATA #REQUIRED>
<!ATTLIST ScaledTileCollection maxscale CDATA #REQUIRED>

```

Each ScaledTileCollection is build by 1 : n image files. This makes it possible to tile large sized maps for optimizing the access of the WMS. Only required tiles will be loaded.

```

<!ELEMENT Tile (#PCDATA)>
<!ATTLIST Tile name CDATA #REQUIRED>
<!ATTLIST Tile minx CDATA #REQUIRED>
<!ATTLIST Tile miny CDATA #REQUIRED>
<!ATTLIST Tile maxx CDATA #REQUIRED>
<!ATTLIST Tile maxy CDATA #REQUIRED>

```

Each tile is described by its name and its bounding box. The coordinate reference system used for the tile-boundingbox must be identical to the one defined for the whole layer.

The <Preview> tag of the layers makes it possible to reference an overview map for a quick visualization of large areas.

Example for a layer definition:

```
<GCLayer name="world" minx="-20.9" miny="37" maxx="35.7"
maxy="70"
  cs="EPSG:4326">
  <Preview>
    <Tile name="preview"
      minx="-20.9" miny="37" maxx="35.7" maxy="70">
      file:///deegree_home/data/raster/world_new_3.jpg
    </Tile>
  </Preview>
  <ScaledTileCollection minscale="0" maxscale="40000">
    <Tile name="level1"
      minx="-20.9" miny="37" maxx="35.7" maxy="70">
      file:///deegree_home/data/raster/level1_4_2.gif
    </Tile>
    <Property name="projekt" value="TestIt"/>
  </ScaledTileCollection>
  <Property name="ID" value="#212"/>
</GCLayer>
```

1.1.1.5. Registration of ESRI-Shape-Files as Data Sources (in deegree)

Similar to the process of reading image data is the registration of a class for connection to ESRI ShapeFiles. In the default setting, the following class is responsible for this:

org.deegree_impl.services.wms.filterservice.ShapeFilter

Subsequently, the definition of layer and data source are necessary. The declaration of an own configuration file is not essential.

```
Class
name="org.deegree_impl.services.wms.filterservice.CachedShapeFilter">
  <Layer name="WorldBorder">

    <KnownDataSource>$deegree_home/data/shape/country</KnownDataSource>
  </Layer>
  <Layer name="cities">

    <KnownDataSource>$deegree_home/data/shape/worldcity</KnownDataSource>
  </Layer>
</Class>
```

The two registered layers are accessible via the WMS by the names "EuroBorder" and "heller". The tag <KnownDataSource> describes the positions of the Shape Files (without

extension). Note that this will be realized by the declaration of the path of the file and not by the declaration of a URL (no `file:///` in front). Similar to the registration of image data, the definition of the Shape File layer dependent on the zoom level is possible.

Note: For the following databases registration, see **deegree** online documentation, Section 3, "Add your own Data to the WMS."

- Registration of an Oracle-(Spatial) Database as data source
- Registration of standard JDBC capable [j2ee]databases for point data as data source
- Registration of other WebMapServers as data source

Appendix 1. deegree WMS Configuration Files

Appendix 1.A. Configuration.dtd

```
<!-- deegree CONFIGURATION DESCRIPTOR
-->
<!--
This DTD defines the structure of the central configuration file
(configuration.xml) of the Dispatcher. It is assumed that not
only WMS servers but also stand alone applications makes use of a
configuration file that's valid to the DTD.
Last Update: July 30, 2002
-->
<!-- root element of the configuration document -->
<!ELEMENT deegreeConfiguration (Capabilities, FilterService*,
DEGService*, RenderService?)>
<!-- defines which capabilities document is accossiated to the
application. -->
<!ELEMENT Capabilities EMPTY>
<!ATTLIST Capabilities
resource CDATA #REQUIRED
>
<!-- defines which filter services are used by the appliction -->
<!ELEMENT FilterService (FeatureInfoFormat*, Class+)>
<!-- defines the deg service used by the appliction -->
<!ELEMENT DEGService (Class)>
<!-- defines the render service used by the appliction -->
<!ELEMENT RenderService (Class)>
<!-- defines the legendservice used by the appliction -->
<!ELEMENT LegendService (Class+)>
<!-- definies the xslt-stylesheets used for processing the
internal xml-documents containing the result to an
feature info request -->
<!ELEMENT FeatureInfoFormat EMPTY>
<!ATTLIST FeatureInfoFormat
format CDATA #REQUIRED
xsltsource CDATA #REQUIRED
>
<!-- defining name a known datasources of a special filter
service.
using the Config tag a additional configuration file (not based
on
this dtd) can be definied for each filter service -->
<!ELEMENT Class (Config?, Layer*, Style*)>
```

```

<!-- ATTLIST Class
name CDATA #REQUIRED
>
<!-- a layer includes one or more data sources. each data source
can
be limited to a range of scales using the minscale and maxscale
attributes. each layer is characterized by a name. the name
should
be unique -->
<!-- ELEMENT Layer (KnownDataSource+)
-->
<!-- ATTLIST Layer
name CDATA #REQUIRED
>
<!--
a style element associates a named style to a LegendService
class. If
a getLegend request orders a legend symbol for a style this is
handle
by the LegendHandler class the style is associated to.
-->
<!-- ELEMENT Style EMPTY
-->
<!-- ATTLIST Style
name CDATA #REQUIRED
>
<!-- defines a data source a file or a database table for example
known by a filterservice -->
<!-- ELEMENT KnownDataSource (#PCDATA)
-->
<!-- ATTLIST KnownDataSource
minscale CDATA #IMPLIED
maxscale CDATA #IMPLIED
>
<!-- defines the source of the configuration file for service -->
<!-- ELEMENT Config (#PCDATA)
-->

```

Appendix 1.B. Configuration.xml

```

<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<!-- DOCTYPE deegreeConfiguration SYSTEM "configuration.dtd" -->
<deegreeConfiguration>
<!-- location of the capabilities XML-file -->
<Capabilities
resource="file:///d:/java/source/deegree/xml_files/deegree.xml"/>
<!-- classes that will be registered to the Dispatcher as
services.
Notice that there can be more then one class for each service
that can be registered. For example you may register a class
as filter service for accessing a oracle spatial database
and another for accessing ESRI shape file etc. -->
<FilterService>

```

```

<FeatureInfoFormat format="MIME"
xsltsource="file:///d:/java/source/deegree/xml_files/feature_info
_to_html.xsl"/>
<!-- Each filterservice can enable access to one or more data
sources -->
<!-- filter service to get data from a oracle spatial database --
>
<Class
name="org.deegree_impl.services.wms.filterservice.OracleFilter2">
<!-- file containing specific elements for configuration
the filter service -->
<Config>file:///d:/java/source/deegree/xml_files/oracle.xml</Conf
ig>
<Layer name="SA_FFH_FL">
<KnownDataSource minscale="0"
maxscale="1">ANDREAS.FFH_FL</KnownDataSource>
<KnownDataSource minscale="1"
maxscale="9999">ANDREAS.GP010100</KnownDataSource>
</Layer>
<Layer name="SA_Parks">
<KnownDataSource>ANDREAS.GP010100</KnownDataSource>
</Layer>
<Layer name="Border">
<KnownDataSource>AGIT.AWSRLDA</KnownDataSource>
</Layer>
<Layer name="River">
<KnownDataSource>AGIT.AWRIV3ML</KnownDataSource>
</Layer>
</Class>
<!-- filter service to get raster data (grid coverages) -->
<Class
name="org.deegree_impl.services.wms.filterservice.GridCoverageFil
ter">
<Config>file:///d:/java/source/deegree/xml_files/gridcover.xml</C
onfig>
<Layer name="geomis">
<KnownDataSource>geomis</KnownDataSource>
</Layer>
<Layer name="euro">
<KnownDataSource>euro</KnownDataSource>
</Layer>
</Class>
<!-- filter service to get data from a access database -->
<Class
name="org.deegree_impl.services.wms.filterservice.PointDBFilter">
<!-- file containing specific elements for configuration
the filter service -->
<Config>file:///d:/java/source/deegree/xml_files/access.xml</Conf
ig>
<Layer name="Staedte">

```

```

<KnownDataSource>tab_cities</KnownDataSource>
</Layer>
</Class>
<!-- filter service to get data from esri shape files -->
<Class
name="org.deegree.impl.services.wms.filterservice.ShapeFilter">
<Layer name="EuroBorder">
<KnownDataSource>D:/java/source/deegree/shape/EURNUTS0</KnownData
Source>
</Layer>
<Layer name="heller">
<KnownDataSource>D:/java/source/deegree/shape/heller</KnownDataSo
urce>
</Layer>
</Class>
<!-- filter service to get data from other wms compatible servers
realizing a
cascading wms server -->
<Class
name="org.deegree.impl.services.wms.filterservice.CascadeFilter">
<Config>file:///d:/java/source/deegree/xml_files/cascade.xml</Con
fig>
<Layer name="GTOP030:CubeWerx">
<KnownDataSource>http://www.cubewerx.com/demo/cubeserv/cubeserv.c
gi?layers=GTOP030:CubeWerx&styles=default</KnownDataSource>
</Layer>
</Class>
</FilterService>
<!-- defining the deg services to register to the dispatcher.
notice that there is possibly more then one deg service. maybe
you like
to use different implementations for differenrent datasources. -->
<DEGService>
<Class
name="org.deegree.impl.services.wms.degservice.DisplayElementGene
rator_Impl">
<!-- file containing specific elements for configuration
the filter service -->
<Config>file:///d:/java/source/deegree/xml_files/degConfig.xml</C
onfig>
</Class>
</DEGService>
<!-- defining the render services to register to the dispatcher.
notice that there is possibly more then one render service. maybe
you like
to use different implementations for differenrent datasources. -->
<RenderService>
<Class
name="org.deegree.impl.services.wms.renderservice.Renderer_Impl"/
>

```

```
</RenderService>
</degreeConfiguration>
```

Appendix 1.C. Gcdescriptor.dtd

```
<!-- d e e g r e e G R I D C O V E R A G E D E S C R I P T O R
-->
<!--
This dtd defines the structure of the description document for
embedding grid coverage data into a degree based application.
Last Update: July 27, 2002
-->
<!-- root element -->
<!ELEMENT GCDescriptor (GCLayer*)>
<!-- the GCLayer element describes a Grid Coverage Layer. A
GCLayer contains of one or more Attributes that describes the
layer. Since by definition of OGC a Grid Coverage is a Feature
the attributes are part of features properties. each gclayer
contains one or more ScaledTileCollection -->
<!ELEMENT GCLayer (Preview?, ScaledTileCollection+, Property+) >
<!ATTLIST GCLayer name CDATA #REQUIRED>
<!ATTLIST GCLayer minx CDATA #REQUIRED>
<!ATTLIST GCLayer miny CDATA #REQUIRED>
<!ATTLIST GCLayer maxx CDATA #REQUIRED>
<!ATTLIST GCLayer maxy CDATA #REQUIRED>
<!-- coordinate system of the layer -->
<!ATTLIST GCLayer cs CDATA #REQUIRED>
<!-- defines a single tile that contains a preview image for
a layer -->
<!ELEMENT Preview (Tile)>
<!-- a ScaledTileCollection describes properties and tiles
for a defined range of map scales -->
<!ELEMENT ScaledTileCollection (Tile+, Property+)>
<!ATTLIST ScaledTileCollection minscale CDATA #REQUIRED>
<!ATTLIST ScaledTileCollection maxscale CDATA #REQUIRED>
<!-- properties describing the GCLayer and each scaled tile. -->
<!ELEMENT Property EMPTY>
<!ATTLIST Property name CDATA #REQUIRED>
<!ATTLIST Property value CDATA #REQUIRED>
<!-- a tile is the basic part of a gclayer. it names the source
of one image being part of the layer and additional required
parameters for it. the source of the tile should be formatted as
URL -->
<!ELEMENT Tile (#PCDATA)>
<!ATTLIST Tile name CDATA #REQUIRED>
<!-- upper left and lower right corner of the tile in
units of the coordinate system of the tile -->
<!ATTLIST Tile minx CDATA #REQUIRED>
<!ATTLIST Tile miny CDATA #REQUIRED>
```

```
<!ATTLIST Tile maxx CDATA #REQUIRED>
<!ATTLIST Tile maxy CDATA #REQUIRED>
```

Appendix 1.D. Gcdescriptor.xml

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<!--DOCTYPE GCDescriptor SYSTEM "gcdescriptor.dtd"-->
<GCDescriptor>
  <!--
  defines a layer that can be accessed by the deegree-WMS using its
  name "SachsenAnhalt".
  -->
  <GCLayer name="SachsenAnhalt" minx="250000" miny="520000"
  maxx="250000" maxy="520000" cs="EPSG:31492">
    <!--
    defines the image that is used for previewing the layer
    -->
    <Preview>
      <Tile name="preview" minx="250000" miny="520000" maxx="250000"
      maxy="520000">
        file:///d:/java/source/deegree/rasterdata/sachsenanhalt.gif
      </Tile>
    </Preview>
    <!--
    as defined at the GCDescriptor dtd a GCLayer is build from
    several scaled tile collections. the term "scaled" targets the
    validity of the tile collection. The tiles defined within a tile
    collection will only be used if the scale of the requested map is
    between the min- (incl.) and maxscale (excl.) value.
    -->
    <!--
    defines the sources for the layer for a detailed view
    -->
    <ScaledTileCollection minscale="0" maxscale="5000">
      <!--
      tiles (images) building the layer for the scale range of this
      tile collection
      -->
      <Tile name="3834hs" minx="4454323.247" miny="5777898.518"
      maxx="4461523.247" maxy="5785211.0176">
        file:///d:/java/source/deegree/rasterdata/3834hs.tif
      </Tile>
      <Tile name="3934hs" minx="4454220.880" miny="5763132.804"
      maxx="4465474.48" maxy="5774086.307">
        file:///d:/java/source/deegree/rasterdata/3934hs.tif
      </Tile>
      <!--
      properties of the layer that are specific for the scale range of
      this tile collection
      -->
```

```

<Property name="level" value="detail"/>
</ScaledTileCollection>
  <!--
defines the sources for the layer for a general view
-->
<ScaledTileCollection minscale="1" maxscale="9999">

  <!--
tiles (images) building the layer for the scale range of this
tile collection
-->
  <Tile name="sachsenanhalt" minx="250000" miny="520000"
maxx="250000" maxy="520000">
file:///d:/java/source/deegree/rasterdata/sachsenanhalt.gif
  </Tile>
  <!--
properties of the layer that are specific for the scale
range of this tile collection
-->
  <Property name="level" value="overview"/>
</ScaledTileCollection>
  <!--
properties of the layer that are constant for every scale
-->
  <Property name="projekt" value="GISPool"/>
  <Property name="Träger" value="DelphiIMM"/>
</GCLayer>
<GCLayer name="euro" minx="-20.9" miny="37" maxx="35.7" maxy="70"
cs="EPSG:4326">
  <Preview>
  <Tile name="preview" minx="-20.9" miny="37" maxx="35.7"
maxy="70">
file:///d:/java/source/deegree/rasterdata/euro.gif
  </Tile>
  </Preview>
  <ScaledTileCollection minscale="0" maxscale="40000">
  <Tile name="euro" minx="-20.9" miny="37" maxx="35.7" maxy="70">
file:///d:/java/source/deegree/rasterdata/euro.gif
  </Tile>
  </ScaledTileCollection>
  <Property name="projekt" value="TestIt"/>
</GCLayer>
<GCLayer name="geomis" minx="3271910.0634942907"
miny="5206712.5389424879" maxx="3956222.5634942907"
maxy="6113025.038942487" cs="EPSG:4326">
  <Preview>
  <Tile name="preview" minx="3271910.0634942907"
miny="5206712.5389424879" maxx="3956222.5634942907"
maxy="6113025.038942487" >

```

```

nichts
</Tile>
</Preview>
<ScaledTileCollection minscale="400" maxscale="400000000000">
<Tile name="Kachel" minx="3271910.0634942907"
miny="5961972.955609154" maxx="3385962.146827624"
maxy="6113025.038942487" >
file:///D:/java/source/deegree/rasterdata/geomis/brd_geomis_level
0_0_0.jpg
</Tile>
<Tile name="Kachel" minx="3271910.0634942907"
miny="5810920.872275821" maxx="3385962.146827624"
maxy="5961972.955609154" >
file:///D:/java/source/dreegree/rasterdata/geomis/brd_geomis_level
0_0_1.jpg
</Tile>
<Tile name="Kachel" minx="3271910.0634942907"
miny="5659868.788942488" maxx="3385962.146827624"
maxy="5810920.872275821" >
file:///D:/java/source/dreegree/rasterdata/geomis/brd_geomis_level
0_0_2.jpg
</Tile>
<Tile name="Kachel" minx="3271910.0634942907"
miny="5508816.705609155" maxx="3385962.146827624"
maxy="5659868.788942488" >
file:///D:/java/source/dreegree/rasterdata/geomis/brd_geomis_level
0_0_3.jpg
</Tile>

<Tile name="Kachel" minx="3271910.0634942907"
miny="5357764.622275821" maxx="3385962.146827624"
maxy="5508816.705609154" >
file:///D:/java/source/dreegree/rasterdata/geomis/brd_geomis_level
0_0_4.jpg
</Tile>
</ScaledTileCollection>
<Property name="projekt" value="GeoMis Bund"/>
</GCLayer>
</GCDescriptor>

```

Appendix 1.E. jld.dtd

```

<!-- deegree LAYER DESCRIPTOR -->
<!--
The deegreeLayerDescriptor is a sequence of styled layers,
represented at the first level by NamedLayer elements.
-->
<!ELEMENT deegreeLayerDescriptor (NamedStyleCollection*) >
<!-- LAYERS AND STYLES -->
<!--

```


NamedLayer: a NamedLayer uses the 'name' attribute to identify a layer known to the WMS and can contain zero or more NamedStyles. In the absence of any styles the default style for the layer is used.

-->

```
<!ELEMENT NamedStyleCollection (NamedStyle+) >
<!ATTLIST NamedStyleCollection name CDATA #REQUIRED >
<!ENTITY % Symbols "(
PolygonSymbol |
LineStringSymbol |
PointSymbol |
TextSymbol |
ScaledPolygonSymbol |
ScaledLineStringSymbol |
ScaledPointSymbol |
ScaledTextSymbol )" >
<!--
```

NamedStyle: a NamedStyle uses the 'name' attribute to identify a style. A NamedStyle contains one or more Symbols of indentic types.

-->

```
<!ELEMENT NamedStyle ((%Symbols;)+) >
<!ATTLIST NamedStyle name CDATA #REQUIRED >
<!ATTLIST NamedStyle minscale CDATA #IMPLIED >
<!ATTLIST NamedStyle maxscale CDATA #IMPLIED >
<!-- S Y M B O L S -->
<!-- Polygon Symbol -->
<!ELEMENT PolygonSymbol (
Geometry,
FillColor?,
BackgroundColor?,
FillOpacity?,
BackgroundOpacity?,
FillDashMatrix?,
StrokeColor?,
StrokeOpacity?,
StrokeWidth?,
StrokeDashArray? ) >
<!-- Line String Symbol -->
<!ELEMENT LineStringSymbol (
Geometry,
StrokeColor?,
StrokeOpacity?,
StrokeWidth?,
StrokeLineJoin?,
StrokeLineCap?,
StrokeDashArray? ) >
<!-- Point Symbol -->
<!ELEMENT PointSymbol (
```

```

Geometry,
FillColor?,
FillOpacity?,
StrokeColor?,
StrokeOpacity?,
StrokeWidth?,
Mark?,
MarkScale?,
MarkOrientation? )>
<!-- Text Symbol
the text symbol definition is slightly different than the OGC SLD
definition.
the OGC SLD other than deegree doesn't know StrokeColor resp.
StrokeOpacity of a
text; it defines the text color and its opacity using the
fillcolor/opacity tag.
deegree instead uses the fill color to draw the text background
and the stroke
color to draw the text, each with its own opacity.
-->
<!ELEMENT TextSymbol (
Geometry,
Label,
TextPosition?,
FontFamily?,
FillColor?,
FillOpacity?,
FontSize?,
FontStyle?,
StrokeColor?,
StrokeOpacity? )>
<!-- Scaled Polygon Symbol -->
<!ELEMENT ScaledPolygonSymbol (
Geometry,
ScaleAttribute,
LimitedSymbol+) >
<!-- Scaled Line String Symbol -->
<!ELEMENT ScaledLineStringSymbol (
Geometry,
ScaleAttribute,
LimitedSymbol+) >
<!-- Scaled Point Symbol -->
<!ELEMENT ScaledPointSymbol (
Geometry,
ScaleAttribute,
LimitedSymbol+) >
<!-- Scaled Text Symbol -->
<!ELEMENT ScaledTextSymbol (
Geometry,

```

```
ScaleAttribute,  
LimitedSymbol+) >
```

```
<!-- Grid Coverage Symbol -->
```

```
<!ELEMENT GridCoverage (  
Opacity?,  
ColorManipulation?  
)>
```

```
<!-- P A R A M E T E R S -->
```

```
<!--
```

Color parameters, currently colors are RGB encoded using two hex values per channel and prefixed with a hash (#). For example full red is #ff0000.

```
-->
```

```
<!ELEMENT FillColor (#PCDATA)>
```

```
<!ELEMENT StrokeColor (#PCDATA)>
```

```
<!--
```

Opacity parameters, currently opacity is encoded as a double with 0.0 representing transparent and 1.0 representing completely opaque.

```
-->
```

```
<!ELEMENT FillOpacity (#PCDATA)>
```

```
<!ELEMENT StrokeOpacity (#PCDATA)>
```

```
<!--
```

this tag indicates the position in relation to the geometry point where to draw the text

```
-->
```

```
<!ELEMENT TextPosition
```

```
(CENTER |
```

```
LEFT_CENTER |
```

```
RIGHT_CENTER |
```

```
TOP_CENTER |
```

```
BOTTOM_CENTER |
```

```
LEFT_TOP |
```

```
RIGHT_TOP |
```

```
LEFT_BOTTOM |
```

```
RIGHT_BOTTOM |
```

```
Userdefined )>
```

```
<!ELEMENT CENTER EMPTY>
```

```
<!ELEMENT LEFT_CENTER EMPTY>
```

```
<!ELEMENT RIGHT_CENTER EMPTY>
```

```
<!ELEMENT TOP_CENTER EMPTY>
```

```
<!ELEMENT BOTTOM_CENTER EMPTY>
```

```
<!ELEMENT LEFT_TOP EMPTY>
```

```
<!ELEMENT RIGHT_TOP EMPTY>
```

```
<!ELEMENT LEFT_BOTTOM EMPTY>
```

```
<!ELEMENT RIGHT_BOTTOM EMPTY>
```

```
<!ELEMENT Userdefined EMPTY>
```

```
<!ATTLIST Userdefined dX CDATA #REQUIRED >
```

```

<!ATTLIST Userdefined dY CDATA #REQUIRED >
<!ELEMENT FontFamily (#PCDATA)>
<!ELEMENT StrokeWidth (#PCDATA)>
<!ELEMENT FontSize (#PCDATA)>
<!ELEMENT MarkScale (#PCDATA)>
<!ELEMENT MarkOrientation (#PCDATA)>
<!ELEMENT StrokeDashArray (#PCDATA)>
<!ELEMENT FillDashMatrix (#PCDATA | PatternFile)*>
<!--
instead of a FillDashMatrix coded as sequence of '0' and '1' the
FillDashMatrix-Element references a graphic file that contains a
pattern
-->
<!ELEMENT PatternFile EMPTY>
<!ATTLIST PatternFile filename CDATA #REQUIRED>
<!ELEMENT StrokeLineJoin (MITRE | ROUND | BEVEL)>
<!ELEMENT StrokeLineCap (BUTT | ROUND | SQUARE)>
<!ELEMENT MITRE EMPTY>
<!ELEMENT ROUND EMPTY>
<!ELEMENT BEVEL EMPTY>
<!ELEMENT BUTT EMPTY>
<!ELEMENT SQUARE EMPTY>
<!ELEMENT FontStyle (NORMAL | ITALIC | OBLIQUE)>
<!ELEMENT NORMAL EMPTY>
<!ELEMENT ITALIC EMPTY>
<!ELEMENT OBLIQUE EMPTY>
<!-- The mark parameter identifies a mark by name or code. -->
<!ELEMENT Mark (#PCDATA | SymbolFile)*>
<!--
instead of a symbol the Mark-Element references a graphic file
that
contains the symbol
-->
<!ELEMENT SymbolFile EMPTY>
<!ATTLIST SymbolFile filename CDATA #REQUIRED>
<!--
Symbols containing a Geometry tag. The Geometry tag specifies
which
property of which feature of which feature type defines the
geographic
position of a symbol. For that a Geometry tag can contain one
FetchFeatureType tag and one or zero FetchFeature resp.
FetchFeatureProperty tag.
This offers several opportunities. At more than one feature type
within
a layer can be used. Second the specialization of portrayal
rules.
For example you can define a style for all features thats tpe is
"myFType".

```

Then you can specialize the portrayal rules for a serveral named features
 thats type is "myFType".
 At third you can create complex portrayal realiations by combining two or
 more rules for drawing a geometry. For example you can define to draw
 each point which feature type is "myFType" with a large square. Additionally
 each single point (feature) got its own portrayal rule, so its drawn twice.
 It results a map that shows each point as a large square containing an
 individual additional symbol.

```

-->
<!--
Parameter to provide the geometry to be symbolized.
It is most commonly 'fetched' from a named property on the
feature.
-->
<!ELEMENT Geometry
(FetchFeatureType,FetchFeature?,FetchFeatureProperty?) >
<!--
Parameter to provide label content. When label content is
'fetched' from a feature property, the type of the property is
unimportant and the symbol is expected to provide a text version
of the property whatever its type.
-->
<!ELEMENT Label (#PCDATA | FetchFeatureProperty)*>
<!--
Elements indicating that the 'named' property should be fetched
from the feature type / feauter being symbolized.
-->
<!ELEMENT FetchFeatureType EMPTY>
<!ATTLIST FetchFeatureType name CDATA #REQUIRED>
<!ELEMENT FetchFeature EMPTY>
<!ATTLIST FetchFeature name CDATA #REQUIRED>
<!ELEMENT FetchFeatureProperty EMPTY>
<!ATTLIST FetchFeatureProperty name CDATA #REQUIRED>
<!--
defines a symbol that's validity is limited by the values
of the min and max attributes
-->
<!ELEMENT LimitedSymbol (PointSymbol+ | LinestringSymbol+ |
PolygonSymbol+ | TextSymbol+)>
<!ATTLIST LimitedSymbol min CDATA #REQUIRED>
<!ATTLIST LimitedSymbol max CDATA #REQUIRED>
<!--
definies the property thats value is used to define the
LimitedSymbol

```

```

used for rendering
-->
<!ELEMENT ScaleAttribute (#PCDATA | FetchFeatureProperty )*>
<!-- defines the opacity of a grid coverage been part of the
produced map. currently opacity is encoded as a double with 0.0
representing transparent and 1.0 representing completely opaque.
-->
<!ELEMENT Opacity (#PCDATA) >
<!-- enables the manipulation of the color composition of a grid
coverage.
it is assumed the colors are represented in a RGB-color model.
each color
components can be modified increasing or decreasing its value. the
values
of the Red, Green and Blue elements are interpreted as percent to
increase
or decrease each pixel's color value. for example:
<ColorManipulation>
<Red>-50</Red>
<Green>-50</Green>
<Blue>30</Blue>
</ColorManipulation>
this will lead to a decrease of the red and green component of
each pixel by
50% and an increase of the blue component by 30%
-->
<!ELEMENT ColorManipulation (Red?, Green?, Blue?)>
<!ELEMENT Red (#PCDATA)>
<!ELEMENT Green (#PCDATA)>
<!ELEMENT Blue (#PCDATA)>

```

Appendix 1.F. deegree.xml layer descriptor

```

<?xml version="1.0" encoding="UTF-8"?>
<!--DOCTYPE deegreeLayerDescriptor SYSTEM "dld.dtd"-->
<!--
A deegreeLayerDescriptor may contain one or more NamedLayers. In
order to
keep the clarity usually no more than one NamedLayer should not
be defined
within a DeegreeLayerDescriptor. The idea is to define a set of
DeegreeLayerDescriptors
for each project defined at the Deegree WMS.
Each contains one or more NamedStyles. A Style defines the
portrayal rule(s) for one or more Symbols. A symbol is associated
with an exactly one basic geometry-type. So if a style contains
more than one symbol definition to create complex graphical
realisations of the defined properties of the symbol types
must be equal.
Each symbol is assigned to a featuretype-feature-property using

```

the Geometry tag. The FetchFeatureType tag is required. The FetchFeature- and the FeatureFeatureProperty tag are optional. Default

values are FeatureFeatureProperty="geometry" and FeatureFeature="default".

It is possible to specialize the portrayal rules for defined features

and feature properties. For example all properties of a feature that's id

equals "XXXX" and which type is point will be painted

not with the default rule for points but with the complex rule of a NamedStyle that contains a special rule for feature "XXXX".

Last Update: Jan 1, 2001

-->

<deegreeLayerDescriptor>

<!-- this an example layer for the deegree wms server considered are point, linestring (curve), polygon (surface), multipoint, multilinestring (multicurve) and multipolygon (multisurface) geometries.

-->

<NamedStyleCollection name="default">

<!-- do not remove or rename because this is the default portrayal rule for (multi)point features -->

<NamedStyle name="point" minscale="0" maxscale="9999999">

<PointSymbol>

<Geometry>

<FetchFeatureType name="point"/>

<FetchFeature name="default"/>

<FetchFeatureProperty name="geometry"/>

</Geometry>

<FillColor>#00aaFF</FillColor>

<StrokeColor>#111111</StrokeColor>

<StrokeWidth>1.0</StrokeWidth>

<Mark>circle</Mark>

<MarkScale>7.0</MarkScale>

</PointSymbol>

</NamedStyle>

<!-- do not remove or rename because this is the default portrayal rule for (multi)linestring features -->

<NamedStyle name="linestring" minscale="0" maxscale="9999999">

<LineStringSymbol>

<Geometry>

<FetchFeatureType name="curve"/>

<FetchFeature name="default"/>

<FetchFeatureProperty name="geometry"/>

</Geometry>

<StrokeColor>#000000</StrokeColor>

<StrokeOpacity>0.9</StrokeOpacity>

<StrokeWidth>1.0</StrokeWidth>

```

<StrokeLineJoin><MITRE/></StrokeLineJoin>
<StrokeLineCap><BUTT/></StrokeLineCap>
<StrokeDashArray>1</StrokeDashArray>
</LineStringSymbol>
</NamedStyle>
<!-- do not remove or rename because this is the default
protrayal rule for (multi)polygon features -->
<NamedStyle name="polygon" minscale="0" maxscale="9999999">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="surface"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geometry"/>
</Geometry>
<FillColor>#ffff00</FillColor>
<BackgroundColor>#0000ff</BackgroundColor>
<FillOpacity>0.5</FillOpacity>
<!--FillDashMatrix>(1,1,1,0,0,0)
(0,1,1,1,0,0)(0,0,1,1,1,0)(0,0,0,1,1,1)
(1,0,0,0,1,1)(1,1,0,0,0,1)</FillDashMatrix-->
<StrokeColor>#222222</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</NamedStyle>
</NamedStyleCollection>
<!-- style for border features -->
<NamedStyleCollection name="border">
<NamedStyle name="polygon1" minscale="0" maxscale="1">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geom"/>
</Geometry>
<FillColor>#cccccc</FillColor>
<FillOpacity>0.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>5.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</NamedStyle>
<NamedStyle name="polygon2" minscale="0" maxscale="1">
<PolygonSymbol>
<Geometry>

```



```

<FetchFeatureType name="EuroBorder"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geom"/>
</Geometry>
<FillColor>#ff0000</FillColor>
<FillOpacity>0.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#FFFFFF</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</NamedStyle>
<NamedStyle name="polygon3" minscale="1" maxscale="10">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geom"/>
</Geometry>
<FillColor>#ff0000</FillColor>
<BackgroundColor>#ffffff</BackgroundColor>
<FillOpacity>1.0</FillOpacity>
<!--
FillDashMatrix>(1,1,0,0)(1,1,0,0)(0,0,1,1)(0,0,1,1)</FillDashMatr
ix-->
<!--FillDashMatrix>
<PatternFile
filename="file:///d:/java/source/deegree/images/info.jpg"/>
</FillDashMatrix-->
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
</PolygonSymbol>
</NamedStyle>
</NamedStyleCollection>
<!-- style for city features -->
<NamedStyleCollection name="cities"
<NamedStyle name="point">
<PointSymbol>
<Geometry>
<FetchFeatureType name="Staedte"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geom"/>
</Geometry>
<FillColor>#ff0000</FillColor>
<StrokeColor>#000000</StrokeColor>
<StrokeWidth>1.0</StrokeWidth>
<Mark>circle</Mark>

```

```

<!--SymbolFile filename="file:///d:/temp/hearts0a.gif"/-->
<MarkScale>7.0</MarkScale>
</PointSymbol>
</NamedStyle>
<NamedStyle name="texttop">
<TextSymbol>
<Geometry>
<FetchFeatureType name="Staedte"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="GEOM"/>
</Geometry>
<Label>
<FetchFeatureProperty name="CITY_NAME"/>
</Label>
<TextPosition><Userdefined dx="0" dy="-10"/></TextPosition>
<FontFamily>arial</FontFamily>
<FillColor>#dddddd</FillColor>
<FillOpacity>1</FillOpacity>
<FontSize>12</FontSize>
<FontStyle><ITALIC/></FontStyle>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
</TextSymbol>
</NamedStyle>
<!--NamedStyle name="textbottom">
<TextSymbol>
<Geometry>
<FetchFeatureType name="Staedte"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="GEOM"/>
</Geometry>
<Label>
<FetchFeatureProperty name="POP_CLASS"/>
</Label>
<TextPosition><BOTTOM_CENTER/></TextPosition>
<FontFamily>arial</FontFamily>
<FillColor>#dddddd</FillColor>
<FillOpacity>1</FillOpacity>
<FontSize>11</FontSize>
<FontStyle><ITALIC/></FontStyle>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
</TextSymbol>
</NamedStyle-->
</NamedStyleCollection>
<!-- style for water features -->
<NamedStyleCollection name="water"
<NamedStyle name="linestring">

```

```

<LineStringSymbol>
  <Geometry>
    <FetchFeatureType name="River"/>
    <FetchFeature name="default"/>
    <FetchFeatureProperty name="geom"/>
  </Geometry>
  <StrokeColor>#0000FF</StrokeColor>
  <StrokeOpacity>0.9</StrokeOpacity>
  <StrokeWidth>2.0</StrokeWidth>
  <StrokeLineJoin><MITRE/></StrokeLineJoin>
  <StrokeLineCap><BUTT/></StrokeLineCap>
  <StrokeDashArray>1</StrokeDashArray>
</LineStringSymbol>
</NamedStyle>
<NamedStyle name="polygon">
  <PolygonSymbol>
    <Geometry>
      <FetchFeatureType name="River"/>
      <FetchFeature name="default"/>
      <FetchFeatureProperty name="geom"/>
    </Geometry>
    <FillColor>#0000FF</FillColor>
    <FillOpacity>0.5</FillOpacity>
    <FillDashMatrix>1</FillDashMatrix>
    <StrokeColor>#222222</StrokeColor>
    <StrokeOpacity>1.0</StrokeOpacity>
    <StrokeWidth>1.0</StrokeWidth>
    <StrokeDashArray>1</StrokeDashArray>
  </PolygonSymbol>
</NamedStyle>
</NamedStyleCollection>
<!-- polygon style for Sachsen-Anhalt surfaces -->
<NamedStyleCollection name="SAFlaeche">
  <NamedStyle name="polygon">
    <PolygonSymbol>
      <Geometry>
        <FetchFeatureType name="Andreas.FFH_FL"/>
        <FetchFeature name="default"/>
        <FetchFeatureProperty name="GEOM"/>
      </Geometry>
      <FillColor>#00FF00</FillColor>
      <FillOpacity>0.5</FillOpacity>
      <FillDashMatrix>1</FillDashMatrix>
      <StrokeColor>#222222</StrokeColor>
      <StrokeOpacity>1.0</StrokeOpacity>
      <StrokeWidth>1.0</StrokeWidth>
      <StrokeDashArray>1</StrokeDashArray>
    </PolygonSymbol>
  </NamedStyle>
</NamedStyleCollection>

```

```

</NamedStyle>
</NamedStyleCollection>
<!-- Label style for Sachsen-Anhalt map -->
<NamedStyleCollection name="SALabel"
<NamedStyle name="text">
<TextSymbol>
<Geometry>
<FetchFeatureType name="ANDREAS.GP010100"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="GEOM"/>
</Geometry>
<Label>
<FetchFeatureProperty name="NA"/>
</Label>
<TextPosition><Userdefined dx="0" dy="0"/></TextPosition>
<FontFamily>arial</FontFamily>
<FillColor>#dddddd</FillColor>
<FillOpacity>0.5</FillOpacity>
<FontSize>10</FontSize>
<FontStyle><ITALIC/></FontStyle>
<StrokeColor>#002200</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
</TextSymbol>
</NamedStyle>
</NamedStyleCollection>
<NamedStyleCollection name="SC">
<NamedStyle name="scaledpolygon">
<ScaledPolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geometry"/>
</Geometry>
<ScaleAttribute>
<FetchFeatureProperty name="POPDENKM"/>
</ScaleAttribute>
<LimitedSymbol min="0" max="100">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
</Geometry>
<FillColor>#222222</FillColor>
<FillOpacity>1.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>

```

```

</PolygonSymbol>
</LimitedSymbol
<LimitedSymbol min="100" max="200">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
</Geometry>
<FillColor>#666666</FillColor>
<FillOpacity>1.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</LimitedSymbol>
<LimitedSymbol min="200" max="300">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
</Geometry>
<FillColor>#999999</FillColor>
<FillOpacity>1.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</LimitedSymbol>
<LimitedSymbol min="300" max="330">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>
</Geometry>
<FillColor>#cccccc</FillColor>
<FillOpacity>1.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</LimitedSymbol>
<LimitedSymbol min="300" max="200000">
<PolygonSymbol>
<Geometry>
<FetchFeatureType name="EuroBorder"/>

```

```

</Geometry>
<FillColor>#0000ff</FillColor>
<FillOpacity>1.0</FillOpacity>
<FillDashMatrix>1</FillDashMatrix>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
<StrokeWidth>1.0</StrokeWidth>
<StrokeDashArray>1</StrokeDashArray>
</PolygonSymbol>
</LimitedSymbol>
</ScaledPolygonSymbol>
</NamedStyle>
</NamedStyleCollection>
<NamedStyleCollection name="kulaka_c">
<NamedStyle name="point">
<PointSymbol>
<Geometry>
<FetchFeatureType name="KULAKA_C"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geometry"/>
</Geometry>
<FillColor>#ff0000</FillColor>
<StrokeColor>#000000</StrokeColor>
<StrokeWidth>1.0</StrokeWidth>
<Mark>circle</Mark>
<!--SymbolFile filename="file:///d:/temp/hearts0a.gif"/-->
<MarkScale>7.0</MarkScale>
</PointSymbol>
</NamedStyle>
<NamedStyle name="texttop">
<TextSymbol>
<Geometry>
<FetchFeatureType name="KULAKA_C"/>
<FetchFeature name="default"/>
<FetchFeatureProperty name="geometry"/>
</Geometry>
<Label><FetchFeatureProperty name="ID_GEOMETRIE"/></Label>
<TextPosition><Userdefined dx="0" dy="-10"/></TextPosition>
<FontFamily>arial</FontFamily>
<FillColor>#dddddd</FillColor>
<FillOpacity>1</FillOpacity>
<FontSize>12</FontSize>
<FontStyle><ITALIC/></FontStyle>
<StrokeColor>#000000</StrokeColor>
<StrokeOpacity>1.0</StrokeOpacity>
</TextSymbol>
</NamedStyle>
</NamedStyleCollection>

```

```
</degreeLayerDescriptor>
```

Appendix 1.G. XSLT-script for formatting the results of a FeatureInfo Request

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:template match="/">
<HTML>
<HEAD>
</HEAD>
<BODY>
<xsl:apply-templates select="/RootElement/Table"/>
</BODY>
</HTML>
</xsl:template>
<xsl:template match="Table">
<xsl:apply-templates select="Row"/>
</xsl:template>
<xsl:template match="Row">
<TABLE border="1">
<TR>
<TD colspan="2"><xsl:value-of
select="/RootElement/Table/@name"/></TD>
</TR>
<TR BGCOLOR="#0000ff">
<TH>Name</TH>
<TH>Value</TH>
</TR>
<xsl:apply-templates select="Column"/>
</TABLE>
<BR/>
</xsl:template>
<xsl:template match="Column">
<TR BGCOLOR="#CCCCC">
<TD><xsl:value-of select="@name"/></TD>
<TD><xsl:value-of select="@value"/></TD>
</TR>
</xsl:template>
</xsl:stylesheet>
```