

## Table of Contents

What is Mapjunction?.....	1
Installation.....	1
Directory structure of a mapjunction website.....	1
Site configuration files.....	2
Multiple Websites sharing the same data.....	2
Hierarchical Menu Structures.....	2
Layer flat files.....	3
Java Applet.....	5

### Mapjunction Admin Manual

This document is intended for a system administrator that will be creating the mapjunction site.

Greg Cockroft. [greg@agog.com](mailto:greg@agog.com) 9/27/2004. First draft.

### *What is Mapjunction?*

MapJunction is a system for developing web-based GIS applications. It utilizes a modified MapServer for its opengis interface support as the server of map layers. It includes multiple clients for viewing of the data. A java applet based client, a simple javascript client (inlineWMS), and coming soon a flash client.

It relies on geocoder.us for geocoding support.

There are html administration pages to ease the administration of the system and a java applet for uploading content and for georeferencing of raster layers.

### Installation

Follow the instructions in the INSTALL readme in the top directory of source distribution.

To create the structure of the web site you will edit the newsite.sh script and change the paths. After running this script and setting up your web server download the sample site data. <http://www.mapjunction.com/downloads/sampleddata.tar.gz> The build.sh script shows how to add content to a site from a script.

### *Directory structure of a mapjunction website*

admin – html and perl scripts for adminstering map layers

applet – jar and cgi's to support the java applet client.

javascript – example page for the inlineWMS javascript client.

bin – C executables and perl scripts

cgi-view - Perl CGI's to support the clients and holds mapserv.cgi.

config – All the txt configuration files that are unique to the mapjunction site.

downloads – Long term storage for download files like GIS zip files.

dstmaps – Transformed raster layers

files – Short term download area for vrmf and pdf files.

flash – flash client

srcmaps - Vector layers and untransformed raster layers.  
tmp – temporary file area for C and perl programs.

## **Site configuration files**

### **config/site.txt**

This is the main config file for site wide variables. After running newsite.sh you need to edit variables in this file. Lines starting with # are comments. Logos and site dependent information are placed in this file.

## **Multiple Websites sharing the same data**

Sometimes you will have different groups creating the content on a single site, but they would like different branding and different default map layers for their group. All the content for a site is stored in the srcmaps and dstmaps directories. Make a new directory structure for the second site with newsite.sh. Delete the empty srcmaps and dstmaps directories and make symbolic links to the first website. Copy the config directory from the first site to the second site and then modify site.txt,insert.txt,menu.xml files to meet the needs of the second site.

## **Hierarchical Menu Structures**

### **config/menu.xml**

A mapjunction site can contain hundreds of layers. To control the display of these in a hierarchical structure you edit an xml file used to describe the structure. Here is the default file

```
<?xml version="1.0"?>
  <menu name="main" type="DEFAULT"> </menu>
```

All map layers in the system are given a tag. In this case they will all be tagged as default and displayed in a single list. If you wanted to add 2 submenus you could use this.

```
<?xml version="1.0"?>
  <menu name="main" type="DEFAULT">
    <menu name="Aerials" type="AERIAL">
    <menu name="Historic" type="HIST">
  </menu>
```

Now there is a main menu and two sub-menus. The administration webpages will parse this file and an html select is generated in the pages to allow layers to be moved around between menus.

**srcmaps/srcmaplist.map**

Dynamically created mapfile for mapserver from srcmaplist.txt.

**dstmaps/dstmaplist.map**

Dynamically created mapfile for mapserver from multiserverdstlist.cache.txt.

**config/menu.xml**

Small xml file to specify the menu structure in the clients.

**config/searchlist.txt**

List of flat files that get searched from the perl cgis.

**config/serverlist.txt**

List of servers that searched and combined. Not implemented

**config/models.txt**

List of detailed VRML models.

**dstmaps/multiserverdstlist.cache.txt**

Combined dstlist.txt from multiple servers

**config/insert.txt**

Starting information for the java applet. Initial starting location.

**config/passwd.pl**

Password list.

**config/links.txt**

http links that show up in the java applet.

**config/savestate.txt**

Saved url's from the java applet

***Layer flat files***

The descriptions of the installed layers in a mapjunction site are stored in two comma delimited files. srcmaps/srcmaps.txt & dstmaps/dstmaps.txt. The srcmaps file contains an entry for all map layers. The dstmaps file contains an entry for all layers that have been georeferenced. mapserver mapfiles are generated from these files whenever new layers are added. A single source raster could be in dstmaps multiple times. It could be transformed with a multiple methods. You can edit these files directly but typically they are only modified by the perl funtions in bin/findmap.pl

**Format of srcmaps/srcmaps.txt**

#	Column name	data type	description
0	srcname	string	Friendly Name of Layer
1	server	string	Tag for server or "" for local server
2	srcfile	string	path minus .shp or path for quadBlob
3	menu	string	Menu tag
4	viewowner	string	who can view this map
5	downloadowner	string	who can download and print this map
6	type	string	[shape postgis map drawing panoramic photo plate elev]
7	reference	string	[normal base never] When can this be used as coordinate reference to tranform other layers.
8	srcw	int	For rasters the width in pixels
9	srch	int	
10	desiredres	float	Cellsize desired after transform
11	maxres	float	For scale hint
12	uploader	string	Who uploaded the map

### Format of dstmaps/dstmaps.txt

Some of the fields are replicated from srcmaps.txt

#	Column name	data type	description
0	srcname		
1	server	string	name of server that contains layer
2	dstname	string	Friendly Name of Layer
3	dstfile	string	path minus .shp or path for quadBlob
4	menu		
5	viewowner		
6	downloadowner		
7	type		
8	reference		
9	x	float	upper left X These four are in projection coordinates
10	y	float	upper left Y
11	w	float	

#	Column name	data type	description
12	h	float	
13	transform	string	[shift piece 1 <sup>st</sup>  2 <sup>nd</sup>  3rd] Piecewise or which polynomial
14	minres	float	For scale hint. Not used
15	maxres	float	For scale hint Not used
16	cacheres	float	Above this res. Vectors are cached as Image
17	canselct	int	[0 1] Does this layer have attributes to select
18	cansearch	int	[0 1] Should the data for this layer be searched
19	refagainst	string	layer used to georeference this layer.

### ***Java Applet***

The java applet client uses an opengis WMS interface to retrieve map layers. WFS is used to find the location of vector objects that have been picked from a search form. Several parameters are read from the enclosing html page. These specify things like the WMS url and the location of various cgi's that the applet uses. index.pl reads applet.template.html and insert.txt to generate the final page of html that the client browser reads. To set the starting map layers and starting location for the java applet you modify config/insert.txt. The easy way to do this is to get the applet showing exactly what you want. Do a save state. Follow the link to the saved location. Do a show page source in your browser and copy and paste the parameters you need from the page source into insert.txt. You want all parameters from u\_alpha0 to vis4.

The applet has a layer model. This allows things like quick comparisons and blends between layer. 5 Layers are downloaded separately and can be combined locally.