HSLayers mapping framework

Jachym Cepicky[1], Stepan Kafka[1], Premysl Vohnout[2]

[1]Help Service – Remote Sensing s.r.o,
256 01, Benešov, Czech republic
jachym@bnhelp.cz. kafka@email.cz
[2]Czech Centrum for Science and Society,
Radlicka 28
150 00, Praha 5, Czech republic
vohnout@ccss.cz

Abstract. HSLayers is "yet another OpenLayers+ExtJS mapping framework". It is released under GNU/GPL and it is developed under cover of Help Service - Remote Sensing s.r.o. It is targeted towards INSPIRE implementation rules for view services. This paper will introduce HSLayers, it\'s development model and direction of the development. We will introduce new classes, which are extending OpenLayers and ExtJS base classes. The paper will compare HSLayers with other OpenLayers-based mapping frameworks, namely MapFish, which is building on top of same libraries. The examples of current usage of HSLayers will also be presented. The paper is targeted to new users and developers, who would like to use and contribute the development. In the first part, we will describe the history of HSLayers and answer the question, why there is new project instead of joining the existing one. In the next part we will introduce already existing widgets and controls, for example OGC Web Mapping Service (WMS) and Web Feature Service Client, new layer switcher with extended capabilities, printing module, module for distance and length measurment, projection switcher and others. Example of the source code will be included. In the last part we will describe the development model of HSLayers and the way, how to join the development.

Keywords: GIS, OpenLayers, MapFish, ExtJS, JavaScript

Introduction

OpenLayers [1] is great JavaScript library, suitable for creating web-based mapping applications. OpenLayers does not use any server-side component (except for proxy script, which is used for overcoming the Cross-Origin Resource Sharing problem [2]).
ExtJS [3] is JavaScript library, suitable for creating complex web-based applications, with the desktop-like look&feel. Idea of HSLayers is putting this two toolkits together, to create full featured web-based GIS applications. HSLayers is developed mainly by Help Service – Remote Sensing (HSRS), but since it is Open Source Software, the development and users community is getting wider.

History of Web GIS

Soon after the World Wide Web was introduced in the early and mid 1990s, web mapping interfaces started to appear. They typically showed a single map image that was reloaded from a web server after the user clicked to zoom or pan. Panning was implemented either as recentering by clicking on the map image, or by moving one step (half the map size) in one of eight directions (N, NW, W, SW, S, SE, E, NE). The art of adding such pre-2005 web interfaces to pre-web GIS systems using open source software is documented in the book by Tyler Mitchell, Web Mapping Illustrated: Using Open Source GIS Toolkits [4]. Prime names in pre-web open source GIS are MapServer and GDAL/OGR.
In 2005 Google Maps was first announced. This novel web mapping application introduced continuous panning by dragging. The map consisted of many small image "tiles"[1] and the web page used JavaScript to request new tiles as needed (on demand) from the web server, without reloading the entire web page. Zooming in or out still required all tiles to be replaced, but the surrounding web page wasn't reloaded. In addition, the visible map covered more than half of the browser window, in contrast to older web mapping applications with a small map window in a large web page. Google also allowed scripting, so users could put Google's maps on their own websites and mix Google's maps with their own data. [5]

[1] Tiles – this term is used because this images are next to each other like tiles in bathroom.

OpenLayers

OpenLayers is JavaScript toolkit for creating mapping applications in the web browsers. It's development started after Where 2.0 conference by MetaCarta. The scope is to implement open source software equivalent to Google Maps. The first version has been released in June 2006.
OpenLayers is more powerful, than Google Maps toolkit. It has abilities for showing maps, based on various raster and vector formats. It has connectors to many standards and quasi-standards, such as MapServer [6], OGC Web Mapping Service [7], ArcIMS [8] and simple Image layer for raster data, GML [9], GeoRSS [10], KML [11], Text, and others for the vector data and Google [12], Yahoo [13] and VirtualEarth [14] for commercial data providers. The user – creator of mapping application – does not need to take care on differencies between various web browsers and their JavaScript implementation or between various data formats.
As already mentioned, OpenLayers is able to display and edit vector data, based on SVG [15] (Firefox, Opera, Safari) and VML [16] (Internet Explorer) vector graphic format. User can directly edit vectors, provide snapping between various vector features and layers.

ExtJS

ExtJS is a cross-browser JavaScript library for building rich internet applications. It consists from customizable UI[2] widgets, ready to be used by GUI[3]-designers, similar to desktop widgets, which among others are text field and textarea input controls, date fields with a pop-up date-picker, numeric fields, list box, radio and checkbox buttons, wysiwymg html editor, text grids, suitable for spreadsheets, trees, tab panels, toolbars, menus and sliders. Ext was originally built as an add-on library extension of Yahoo UI [17], but now, it is an state-alone project.

HSLayers

HSLayers (Help Service+OpenLayers) [18] combines capabilities of ExtJS and OpenLayers and several helper scripts, to truly Web GIS applications. Development has been started in 2007 and It has been open sourced after 2 years of development, in 2009, and released under conditions of GNU General Public License 3.

Structure

The source of HSLayers consists of two main parts, *patches* and *addons.* Patches are small modifications to original OpenLayers classes. Usually, they are submitted as real patches of existing OpenLayers files and when they are accepted by OpenLayers developers and added to main development tree, they are removed from HSLayers code structure as well. If the modification assigned to HSRS needs only, they will probably stay as patches in HSLayers.
The addons parts can be split again in two parts: Those, which are written with direct support of ExtJS UI, and those, which are OpenLayers-based only.
HSLayers is translated to English, Czech, Latvian and Spanish.

HSLayers components

HSLayers copies the structure of OpenLayers partly, somebody, who knows OpenLayers already, should not be confused, while having a look at HSLayers code.

[2] UI – User interface
[3] GUI – Graphical UI

HSLayers.InfoPanel - Ext.Panel-based class, for displaying various informations from the map, such as results of map queries, search results and others.

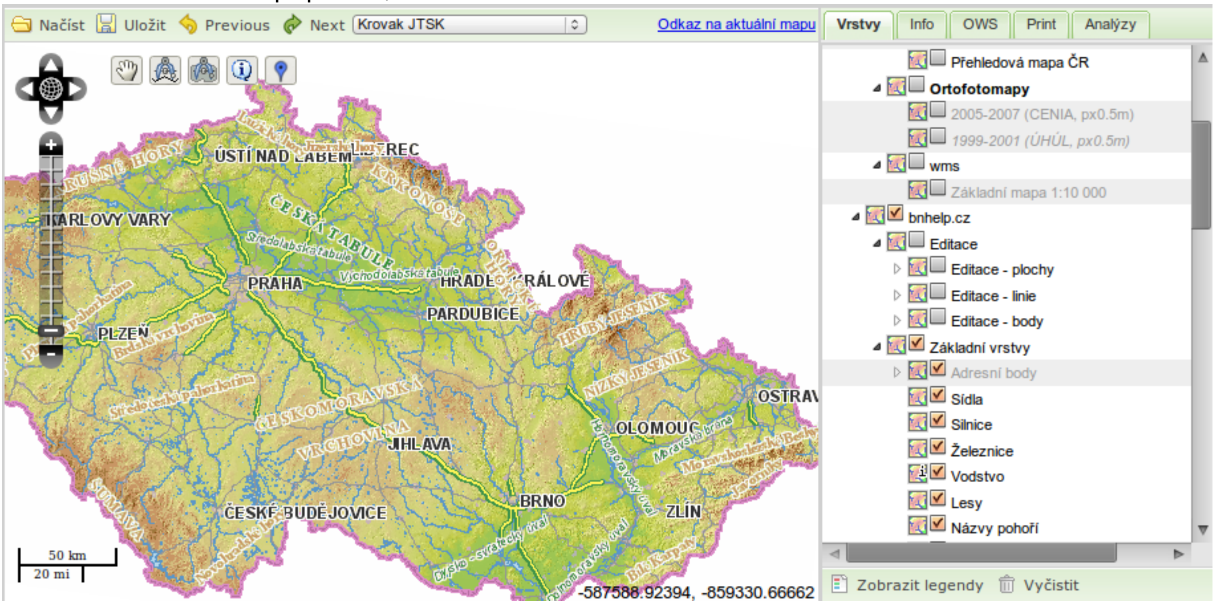


Fig. 1: HSLayers Portal example

HSLayers.MapPanel - Ext.Panel, containting the OpenLayers.Map widget, with several pre-defined controls.

HSLayers.MapPortal - Ready to be used by Portal mapping application, build on top of MapViewer (see next description). Several other fine-tuning features of the MapViewer are predefined here.

HSLayers.MapViewer - Basic map viewer, with map, LayerSwitcher, InfoPanel, OWS clients and Printing control.

HSLayers.OWS – OGC WMS and OGC WFS clients. The client gets input GetCapabilities URL, parses it and displays the layer tree. User can choose several image and query formats and display the selected layers in the map.

HSLayers.Printer - Tool for definition of the printing request. User can set desired scale and paper format, move the paper

HSLayers.Util - Various utils used by applications.

HSLayers – Main Class, used for the namespace definition.

HSLayers.Control.ArgParser - URL argument parser based on OpenLayers.Control.ArgParser, but with additional features, such as support for vector data features encoded into the URL.

HSLayers.Control.BoxLayerSwitcher - Google Maps-like layer switcher.

HSLayers.Control.Click - Base class for custom controls, based on the clicking in the map event.

HSLayers.Control.Cuzk - Tool for getting attributes of the nearest parcel to the point, defined by the user by clicking in the map. The data are parsed from Czech office for survey, mapping and cadaster directly.

HSLayers.Control.DrawControls – Base Class for Point, Line, Polygon and Box drawing controls.

HSLayers.Control.LayerSwitcher – Advanced Layer switcher. This component is described below in this paper.

HSLayers.Control.Measure – Line and area measurement. Several lines or polygons can be drawn to map, the tools displays always sum of areas (length) and area (length) of last feature.

HSLayers.Control.Permalink – This class is based on OpenLayers.Control.Permalink and it is used for storing aditional informations, such as vector features, in the URL.

HSLayers.Control.Pin - Tool for adding the marker with description in the map. The information about the marker is added to Permalink, and can be send to other user.

HSLayers.Control.ProjectionSwitcher – The map can be switched to different coordinate system.

HSLayers.Control.Query – Tool for interactive point and box queries. The tool is able to form the WMS or MapServer feature request and parse and display several types of responses, namely GML, ArcIMS GML, text and HTML.

HSLayers.Control.State – Will save and restore the workspace of running application for later reusage.

HSLayers.Layer.ChartLayer – Vector layer, which is able to display chart diagrams, based of features attributes.

HSLayers.Layer.MapServer – While OpenLayers.Layer.MapServer provides just the simple interface to proprietary UMN MapServer format, the version in HSLayers is more complex. It includes the complete structure of the mapfile (MapServer configuration file) and so, the content of the OpenLayers.Layer can be changed. This component will be described in more detail later in this paper.

HSLayers.Layer.WarpedWMS – this class is used, when the WMS server does not support the displayed map projection. The images are warped with help of small server-side script.

HSLayers.Layer.WFSSensor – Layer for displaying the sensor informations through OGC WFS.

We will describe some few classes more deaily.

HSLayers.Layer.MapServer

One of the features of the OpenLayers concept is, that each map layer, represented by the OpenLayers.Layer class object, is represented as grid of tiled images. The content of the image is set while the layer is initialized and there is no easy way, how to change the layers content.

Example 1. Definition of the OpenLayers.Layer.MapServer instance

```
var new_layer = new OpenLayers.Layer.MapServer(„Layer's Name “,
                    “http://foo.bar “,
                   {layers: „streets landuse “}, {});
```

In the previous example , new OpenLayers.Layer.MapServer instance is created and it should contain two layers, defined in the configuration file (mapfile), by name “streets” and “landuse”. Currently, there is no easy way, how to change this, using any method in OpenLayers.Layer.MapServer. And there is also no way, how to pass the information about the mapfile structure to OpenLayers. This is big disadvantage, if we need to display larger amount of data, and in the same time, we need to have control of the map content. Every mapfile layer would have to be defined as single OpenLayers.Layer class, so the stress of the client would get bigger, as well as the server stress, which can not render the informations in one image, but it has to render separate image for every layer separately. The class HSLayers.Layer.MapServer tries to solve this.

Example 2. Definition of HSLayers.Layer.MapServer instance.

```
var new_layer = new HSLayers.Layer.MapServer(„Layer's Name “,
                    “http://foo.bar?mode=lyrlist “,
                   {}, {});
```

The mode=lyrlist is optional and depends on the server configuration. This given address should return JSON [18] structure of the mapfile. The layer and layer group hierarchy is created and stored in groups attribute of the class from JSON structure, as reprint of the mapfile. This attributes can be processed later by the layer switcher.

Example 3. MapFile structure returned by the server, which is used for HSLayers.Layer.MapServer (part)

```
{
       "name":"editace",
       "title":"Editace",
       "layers":[
              {
                     "name":"plochy",
                     "title":"Editace - plochy",
                     "visible":0,
                     "minScale":-1,
                     "maxScale":-1,
                     "legendURL":"http://bnhelp.cz/mapserv/hsmap/hsmap.php?¥
                               project=cr_hslayers&mode=legend&layers=plochy",
                     "metadataURL":"",
                     "switchers":[""],
                     "queryable":0,
                     "edit":3,"snap":0
              },
              {
                     "name":"linie",
                     "title":"Editace - linie",
                     "visible":0,"minScale":-1,"maxScale":-1
                     …
              }
       ]
},
{
        "name" :  "Topo" ,
        …
},
...
```

HSLayers.Control.LayerSwitcher

As part of the development, we had to work on layer switcher, which will be possibly as functional, as the users are used to from common desktop GIS. HSLayers.Control.LayerSwitcher is able to display layers in hierarchical structure. It introduces new attribute of the layer – group – which contains the title displayed in the tree structure. Name of the layer is taken, if there is no group attribute of the layer. Layers are merged into one tree node, if there are several layers with same group name (or name)., Layers will be switched on/off without any further action from user, if they have min/maxScale attribute set. That's, why the group attribute of each layer is introduced – one tree node in the layer switcher represents one or more OpenLayer.Layer instances.
Another feature of the layer switcher is the capability of creation of hierarchical structures. Several nodes will be created, if the group name contains slash ("/") mark. The layer switcher creates hierarchical structures based on previously described HSLayers.Layer.MapServer attributes.
Drag'n'drop capabilities is another feature of layer switcher. This feature allows user to customize the layer order and menu, with Source URL, Metadata URL, opacity settings and SLD settings.

Example 4. Name definition, converted by the layer switcher to tree hierarchy

```
var new_layer = new OpenLayers.Layer.WMS( "Name" ," http://foo/bar/wms" ,
                              {group:  "Základní vrstvy/Adresní body" },{});
```

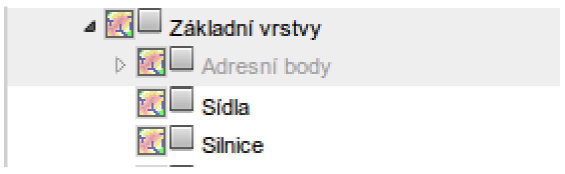


***Fig. 2**: Result of example definition of the group name and it's conversion to tree hierarchy*

Server scripts

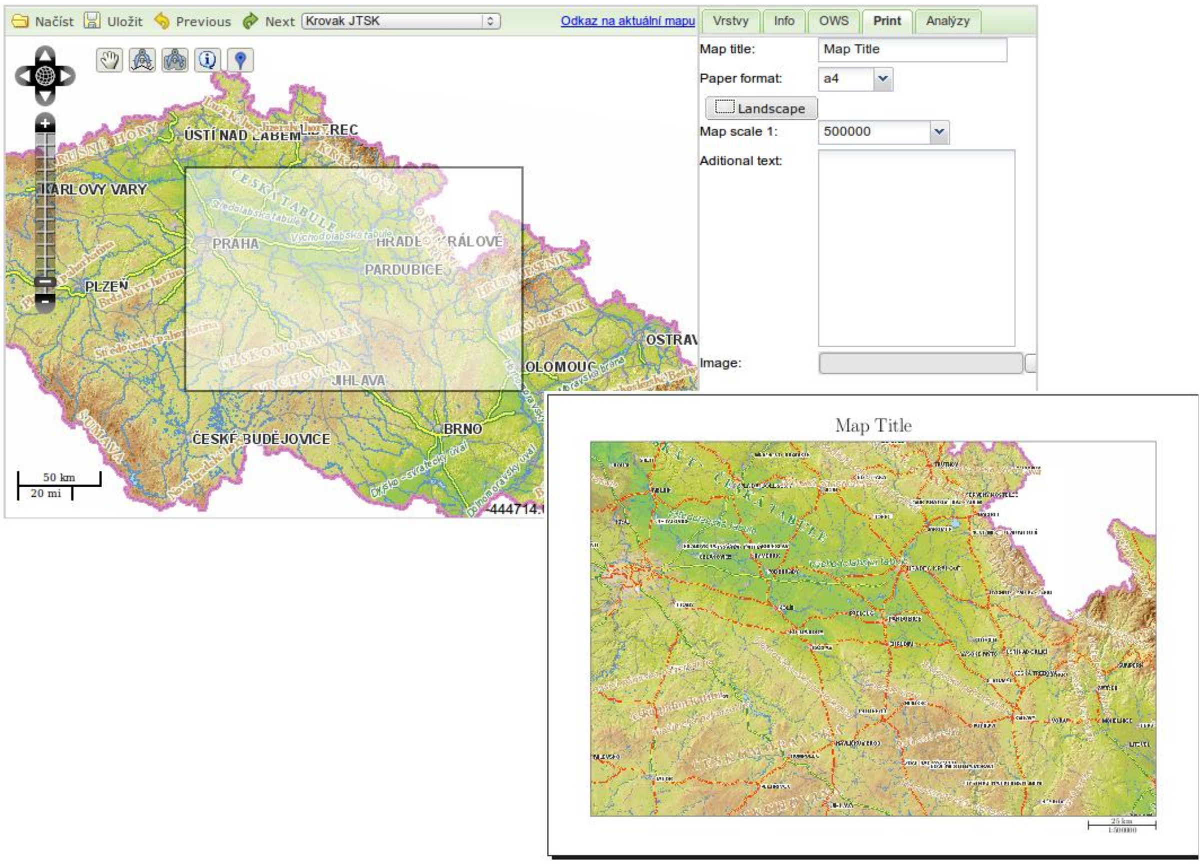
Several server-side scripts are necessary if we need to use all of the features of HSLayers. They are located in the source/scripts directory of the source code.

· **feedback.php** is used by the load/save state control. The workspace state is stored with help of php session id attribute and when the user comes back on the site, the application will be loaded in the last visited state.

· **MapCreator.py** is printing tool. It gets JSON structure with map attributes (scale, title, …) and list of layers with specified tiles as input. All images are downloaded, positioned to printing canvas and stored as PDF file, which is then returned back to the user.

· **Warper.py** – Script for warping given WMS GetMap request url to specified coordinate system. Python bindigs of GDAL/OGR and PROJ.4 libraries are used for the transformation.

However, the user does not need to have those scripts and still she gets the full featured mapping application with WMS and WFS clients.



Map title: Map Title
Paper format: a4
Map scale 1: 500000
Aditional text:

Fig. 3: Printing module.

Other OpenLayers and ExtJS implementations

MapFish [21] is similar framework, using OpenLayers and ExtJS combination. MapFish uses server-side components, they were combination of Python and Java previously, nowadays, they are Python-only. The concept of MapFish is a little bit different, compared to HSLayers. MapFish builds more on ExtJS libraries and creates new classes with embed OpenLayers components. HSLayers tries to build more on top of OpenLayers. Anyway, it is possible to combine components from both and users are doing so.

GeoExt [22] is the newest one of so far known toolkits. It is strictly JavaScript oriented and it is based on ExtJS completely. GeoExt reimplements some OpenLayers controls in ExtJS as well (for example the PanZoomBar control).

None of those toolkits was ready for HSRS daily usage. That was one of the reasons, why we started to work on our custom solution. However, it is possible in the future, that we join some of the existing projects.

Licensing

HSLayers is released under conditions of GNU/GPL v.3. However, since HSRS is in several projects with partners from different countries, this license is not acceptable for some of them. HSRS decided to use dual licensing model, similar to some other Open Source projects. Since HSRS will be the copyright holder of the source code (this does not affect the authorship of the code), we are able to offer special licensing conditions to our partners. However, we guarantee also, that the code remains Free Software.

Links and downloads

HSLayers project page is located at http://dev.bnhelp.cz/trac/hslayers. Currently (November 2009), it is available in 1.3.0 stable version. The new 2.0.0 version is in beta stage and we hope to release it in next weeks. HSLayers can be downloaded from the project download page, as well as from subversion repository directly, using svn://bnhelp.cz/hslayers address.

Conclusion

HSLayers is stable set of tools for creation of WebGIS applications. It is distributed and developed as Free Software under conditions of GNU/GPL v3 since version 1.3.0. Version 2.0.0 will be released In several weeks. There are some affords to improve original projects by HSLayers. OpenLayers has mainly accepted several patches. HSLayers did not join any existing projects, like MapFish or GeoExt, because of their different approach and philosophy, also the copyright holding is essential for HSRS business model. HSLayers development team is encouraging other developers and users to help them with the development.

Reference

[1] Author. Title of the paper in the proceedings or journal Name and year of a conference or name, year and issue of a journal (in italics). Place and year of issue. ISBN.

[1] Metacarta, *OpenLayers*, on-line http://openlayers.org, 2009-11-11

[2] David Ondřich, *Cross-Site XMLHttpRequest*, Zdrojak, 2009, on-line http://zdrojak.root.cz/clanky/cross-site-xmlhttprequest/ , ISSN: 1803-5620

[3] *ExtJS*, on-line http://extjs.com, 2009-11-11

[4] Tyler Mitchell. *Web Mapping Illustrated* , O'Reilly Media, 2005, ISBN: 978-0-596-00865-9

[5] *OpenLayers History*, on-line http://trac.openlayers.org/wiki/History 2009-11-11

[6] *UMN MapServer, on-line* http://mapserver.umn.gis.edu*, 2009-11-11*

[7] Open Geospatial Consortium. *OGC Web Mapping Service Interface.* Jeff de la Beaujardiere, editor. OGC 03-109r1, 2004, on-line http://opengeospatial.org/standards/wms 2009-11-11

[8] Esri, *ArcIMS,* on-line http://www.esri.com/software/arcgis/arcims/ 2009-11-11

[9] Open Geospatial Consortium. *OGC Geography Markup Language (GML) Encoding Standard* , Clemens Portele, editor. OGC 07-036, 2008, on-line http://opengeospatial.org/standards/gml, 2009-11-11

[10] *GeoRSS, on-line* http://georss.org *2009-11-11*

[11] Open Geospatial Consortium. *OGC KML*, Tim Wilson, editor. OGC 07-147r2, 2008, on-line http://opengeospatial.org/standards/kml 2009-11-11

[12] Google Inc. *Google Maps API*, on-line http://code.google.com/intl/cs/apis/maps/ 2009-11-11

[13] Yahoo! Inc. *Yahoo! Maps Webservices,* on-line http://developer.yahoo.com/maps/ 2009-11-11

[14] Microsoft Inc. *Bing Maps Interactive SDK,* on-line http://www.microsoft.com/maps/isdk/ajax/ 2009-11-11

[15] W3C. *Scalable Vector Graphics (SVG) 1.1 Specification,* Jon Ferraiolo, FUJISAWA Jun, Dean Jackson, editors. On-line http://www.w3.org/TR/SVG11/ 2009-11-11

[16] W3C, *Vector Markup Language (VML)*, on-line http://www.w3.org/TR/NOTE-VML 2009-11-11

[17] Yahoo! Inc. *YUI Library,* on-line http://developer.yahoo.com/yui/ 2009-11-11

[18] *JSON,* on-line http://json.org/ 2009-11-11