# Progetto

Creato progetto CartoWebNet2 su parodis-dts-pc

## Struttura directory

* **CartoWebNET2**
  + **config**: contiene i file json di configurazione
  + **img**: contiene le immagini
    - legend: immaigni legenda statiche
  + **doc**: documentazione associata al progetto
  + **lib**: contiene le librerie javascript
    - **Cwn2**: libreria cwn2
      * **apidocs**: contiene la documentazione delle api js
      * **src**
        + **buttons**
        + **widgets**
      * **test**: contiene le unit test
        + **config**
      * **apidocs**
    - **ExtJS**
    - **ExtJS-ux**:librerie utente ExtJS
    - **GeoExt**
    - **OpenLayer**
  + **services**: servizi lato server (contenente i progetti .NET)
  + **stili**: contiene gli stili
    - **default** (stile di default)
      * **icons**
  + **templates**: contiene dei template html/js per vari casi
  + **tools**: contiene vari strumenti

## Librerie Javascript

**Openlayers**: Installata versione dev di Openlayers su E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib (risolve problemi popup "dati mappa" google)

* creato repository da <http://svn.openlayers.org/trunk/openlayers/>
* scaricato <http://openlayers.org/dev/OpenLayers.js>

**ExtJS**: Installato extJs 3.3.3 su E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib (url <http://dev.sencha.com/deploy/ext-3.2.1.zip>)

**Geoext**: Installato geoext 1.0 su E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib

**Json**: Installato json E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib

**uxmedia.js:** Installata libreria per gestione file pdf/html/png/jpg in finestra ExtJS (installato in directory lib/ExtJS-ux

## IDE: WebStorm

Help

<http://www.jetbrains.com/webstorm/webhelp>

### Modifiche Preferenze

* Inspections
  + eliminato inutili
* Javascript libraries definito
  + <http://blog.jetbrains.com/webide/2010/11/working-with-javascript-libraries-in-phpstorm-webstorm/>
  + jQuery
  + OpenLayers
  + ExtJS
  + GeoExt
* Definito External Tools
  + Build CWN2
  + ApiDocs CWN2 (ALT+A)
  + Build CWN2 Produzione
* Keymap
  + Sostituito Ctrl+Y con azione undo
* Plugins
  + disabilitato quelli non utilizzato
  + installato Scratch

### Macro

Build CWN2 (ALT+B): save all files + Build CWN2

## Tool per BUILD libreria JS Cwn2

**ANT**

<http://www.julienlecomte.net/blog/2007/09/16/>

<http://www.javascriptr.com/2009/07/21/setting-up-a-javascript-build-process/>

Scaricato ANT 1.8.2 in CartoWebNET2/tools

Copiato yuicompressor-2.4.2 in CartoWebNET2/tools

Creato file di configurazione ANT: src/build.xml

Modificato build.bat

## Tool per Documentazione API JS

Utilizzo NaturalDocs <http://www.naturaldocs.org/>

Richiede Perl (installato in C:\perl)

Per eseguire lanciare E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\tools\naturaldocs\nd.bat

Oppure utilizzare il comando in WebStorm

Documentazione viene creata in E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib\Cwn2\apidocs\docs\index.html

Url documentazione:

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/lib/Cwn2/apidocs/docs>/index.html

# Definizione stile Css per tool base Openlayer

Vedi

* <http://docs.openlayers.org/library/controls.html#styling-panels>
* <http://support.mapbox.com/kb/mapping-101/openlayers-themes>

Ho sostituito le immagini con quelle di mapbox. L imagini sono state copiate in E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib\Openlayers\img e E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\lib\Openlayers\theme\default\img. Ho fatto un backup delle dir sotto: E:\progetti\[CartoWebNet2](http://code.google.com/p/cartowebnet2/wiki/CartoWebNet2)\backup img openlayers mapbox. Quando faccio un upgrade di ol devo ricopiarle nelle directory. Escluso l'immagine blank.gif che causa problemi di trasparenza in IE (ripristinata l'immagine originale)

# Bottoni Navigazione

Realizzato bottoni di navigazione

* Pan
* Zoomin
* Zoomout
* Fit
* Previous
* Next

Codice:

* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\pan.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\zoomin.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\zoomout.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\fitall.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\zoomprevious.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\zoomnext.js

# Bottoni Misurazione

Realizzato bottoni di misurazione.

* Misurazioni lineari
* Misurazioni areali

L’output compare sulla statusbar.

Codice:

* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\measurearea.js
* E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\measureline.js

# Aggiunta Livelli

Realizzato bottone di aggiunta livelli che apre finestra di prova contenente la url che punta ad un file di configurazione JSON che contiene l’array dei livelli da caricare.

Codice: E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\loadlayers.js

# Eliminazione Livelli

Realizzato bottoni di eliminazione livelli. Apre una finestra con elenco dei livelli e check-box per selezione.

Codice: E:\progetti\CartoWebNet2\lib\Cwn2\src\buttons\removeLayers.js

# Gestione Trasparenza

Realizzato bottone per gestione trasparenza.

Codice: buttons/transaprency.js

## Problema Trasparenza su vettoriali con attivo il controllo FeatureSelect

Trasparenza non funziona su layer vettoriali se è attivo il control FeatureSelect attivo su + di un layer vettoriale.

Baco OpenLayers, vedi <http://osgeo-org.1803224.n2.nabble.com/OpenLayer-Select-Feature-and-Root-Container-td6382489.html>

Risolto con override del metodo Layer.setOpacity (in OpenLayersExt.js)

# Ingresso con filtro su uno o più oggetti vettoriali

Permette di rappresentare oggetti cartografici dinamicamente all’avvio della applicazione.

Viene definito nella configurazione un layer vuoto, ovvero con la proprietà “loadVoid” impostata a true, e protocollo in formato “GeoJSON”.

Viene quindi passato alla applicazione un oggetto GeoJSON contenente gli oggetti da rappresentare

L’oggetto viene definito nel codice js in una variabile e caricato sul layer con il metodo “loadGeoJSON” (definito in OpenLayersExt.js) a cui viene passata la variabile contenente il GeoJSON.

Vedi:

* E:\progetti\CartoWebNet2\MappaCultura.html
* E:\progetti\CartoWebNet2\MappaCultura.js
* E:\progetti\CartoWebNet2\config\MappaCulturaConfig.json

# Ingresso con evidenziazione di uno o più oggetti vettoriali

Permette di evidenziare una serie di oggetti appartenenti ad uno o più layer vettoriali.

## Evidenziazione

Gli oggetti vengono evidenziati applicando lo stile “hilite” della stylemap.

<http://trac.osgeo.org/openlayers/wiki/Styles>

Esiste un controllo OL di tipo SelectFeature (CWN2.Base.hilite\_feature\_control) con solo hilite per gestire l’evidenziazione di feature in operazioni di filtro/find/query.

Viene creato in MapManager.activateSelectFeatureControl

**Evidenziazione all’avvio della applicazione**

E’ necessario configurare la sezione “hilite” all’interno della configurazione del layer contenente:

* layerName: nome del layer
* attrName: nome dell’attributo su cui effettuare la selezione
* items: array dei valori
* options: opzioni varie
  + zoom (true/false): indica se deve essere effettuato lo zoom

Per ogni livello vettoriale l’evento “featuresadded” che richiama la funzione CWN2.Callbacks.onFeaturesAdded

Se il livello ha la sezione “hilite” configurata viene richiamata la funzione: OpenLayers.Layer.Vector.prototype.hiliteFeatures (in OpenLayersExt.js) che effettua l’hilite delle features e carica l’array CWN2.Base.selectedFeatures.

Per gestire hilite insieme a select interattiva ho fatto un override del metodo OpenLayers.Control.SelectFeature.prototype.unhighlight (in OpenLayersExt.js).

**Applicazione integrata in una pagina con comunicazione via javascript**

Vedi esempio EsempioWindow.html

Il codice javascript della pagina deve chiamare la funzione CWN2.Util.hiliteFeature passandogli un oggetto di configurazione dell’hilite strutturato coma la sezione hilite della configurazione del layer.

# Label su livelli vettoriali

Il nome dell’attributo dell’oggetto che contiene il valore da visualizzare viene definito nella configurazione mediante l’attributo:

layers[n].styleMap.graphicTitle

Questo attributo contiene il nome di una delle “properties” del GeoJSON

Vedi:

* E:\progetti\CartoWebNet2\MappaCultura.html
* E:\progetti\CartoWebNet2\config\MappaCulturaConfig.json

# Info su livelli vettoriali

La info sui livelli vettoriali prevede due livelli di configurazione:

* statica
* dinamica

Il nome dell’attributo dell’oggetto che contiene il valore da visualizzare viene definito nella configurazione mediante l’attributo:

layers[n].styleMap.graphicTitle

Questo attributo contiene il nome di una delle “properties” del GeoJSON

Vedi:

* E:\progetti\CartoWebNet2\MappaCultura.html
* E:\progetti\CartoWebNet2\config\MappaCulturaConfig.json

## Problema popup dopo aggiunta livelli vettoriali

Se aggiungo un layer vettoriale mi si sputtana la gestione degli eventi le pop-up non vengono cancellate sull’unselect, e non funziona il tasto close.

Risolto con un HACK: sulla funzione onFeatureSelect (CallBacks.js)

## Problema selezione su livelli vettoriali dopo aggiunta di altri livelli vettoriali

Se aggiungo un layer vettoriale mi si sputtana la gestione degli eventi (non riesco ad interrogare i layer vettoriali presenti precedentemente sulla mappa).

Risolto re-inizializzando i controlli openlayer

# Legenda

## Infrastruttura

Sviluppata l’infrastruttura generica per la visualizzazione della legenda. La legenda viene rappresentata su un panel collassabile nella regione est.

Codice:

* E:\progetti\CartoWebNet2\lib\Cwn2\src\LegendPanelBuilder.js

## Legenda Semplice

Codice: Funzione “simple” dell’oggetto LegendPanelBuilder

Implementata la legenda di tipo “semplice” contenente:

* Livelli di base
* Livelli di overlay

E’ visibile un solo livello di base alla volta

Vengono visualizzate:

* icona
* nome del livello
* check-box per controllo visibilità per i livelli di overlay
* radio-button per controllo visibilità per i livelli di base

## Gestione Finestra PopUp

Nel caso opportunamente configurata la legenda permette, mediante la selezione dell’icona del layer, l’apertura di una pagina contente , a seconda della configurazione:

* un documento (pdf/html/png/jpg) se nella configurazione del layer sia impostata la proprietà legend.legendUrl contenente la url di un documento
* la rappresentazione di dettaglio della vestizione grafica nel caso sia impostata la proprietà legend.classes contenente un array di oggetti con le seguienti proprietà:
  + label
  + icon

CODICE:

* E:\progetti\CartoWebNet2\lib\Cwn2\src\LegendWindow.js

NOTA:

Modificato stile E:\progetti\CartoWebNet2\lib\ExtJs\docs\resources\css\ext-all.css

Elemento .x-grid3-cell-inner cambiato white-space da nowrap a normal

In questo modo la label della legenda va a capo (wrap del testo)

# Aggiunta livelli

Realizzata infrastruttura js che permette l’aggiunta di livelli (vedi addLayers in Configuration.js)

Proxy: installato OLProxy.py in cartowebnet2/services/

Realizzato Bottone di aggiunta livelli

CODICE: buttons/loadlayer.js

Implementata la funzione per il caricamento di layer per canali tematici

La configurazione del bottone prevede le seguenti opzioni:

**panels**: array contenente i pannelli da caricare, al momento “canali”

**canaliTematiciTreeServiceUrl:** url del servizio che ritorna la struttura del tree

**getLayersConfigServiceUrl:** url del servizio che ritorna la configurazione dei layer da caricare

# Ricerche

Bottone che permette di effettuare ricerche sulla mappa.

Le funzioni di ricerca sono raggruppate in una finestra multitab.

Permettono di effettuare ricerche per:

* località
* indirizzo (utilizzando servizio google)

In fase di configurazione si può decidere quali funzioni attivare

Il punto trovato viene evidenziato con un segnapunto. E viene fatta uno zoom nell’intorno

Il codice è in buttons/fins.js

## Ricerca per indirizzo

Effettua una ricerca utilizzando i servizi di geocoding di Google.

Viene presentata una combo dinamica con auto-completamento

Documentazione Google:

v2 <http://code.google.com/intl/it-IT/apis/maps/documentation/geocoding/v2/>

v3 <http://code.google.com/intl/it-IT/apis/maps/documentation/geocoding/index.html>

**PROXY**

Creato Proxy per accedere ai servizi Google. Il proxy si trova in: E:\progetti\CartoWebNet2\services\proxy\googleProxyServer.asp

La url del servizio è

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/services/proxy/googleProxyServer.asp?ll=44,8.5&spn=1,1.5&hl=it&gl=it&sensor=false>

Ho fatto anche un proxy per I servizi Google V3

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/services/proxy/googleProxyServerV3.asp?region=it&sensor=false&bounds=43,7|45,10>

Il proxy testa il server su cui si trova

* se il server è sul dominio “datasiel.net” viene richiamato il proxy che si trova su RCARTO3 al seguente indirizzo <http://www.cartografiarl.regione.liguria.it/CartoWebNET/googleProxyServer.asp?> Che a sua volta richiama il servizio google
* altrimenti viene richiamato direttamente il servizio google "http://maps.google.com/maps/geo?"

CODICE:

Implementato codice in buttons/find.js

Utilizza Google API v2 (problemi con V3: codice in findAddressPanelV3

## Ricerca per località

Effettua una ricerca sulla banca dati Regionale dei toponimi

Nel caso di ricerca località viene caricato un layer con i toponimi (vettoriale o WMS)

## Configurazione

La configurazione del bottone prevede le seguenti opzioni:

**panels**: array contenente i pannelli da caricare, al momento “indirizzo”,”toponimo”

**addressGeocodeServiceUrl:** url del servizio di geocoding Google

**toponimiServiceUrl:** url del servizio che ritorna un json

**flagLimitaTerritorioLigure (true/false):** se impostato a true viene controllato che il punto ritornato ricada nel territorio ligure

# Server GeoJSON

Sviluppato un servizio in asp.net nell’ambito di cwn2 (services/GeoJSON) che utilizza la funzione Oracle SDO\_UTIL.TO\_WKTGEOMETRY(geometry) e la converte in GeoJSON utilizzando i metodi per serializzare e deserializzare in WKT e GeoJSON di UtilityCarto. Vengono gestite le seguenti geometrie:

* POINT
* LINESTRING
* POLYGON
* MULTILINSTRING
* MULTIPOLYGON

# Prove servizi WFS

Effettuato test con client OL e servizi WFS MapServer.

**PROBLEMA**

MapServer ha problemi a gestire le richieste in POST (non viene mandata la geometria), quindi l'utilizzo di layers di tipo Openlayers.Vector con formato WFS non funziona.

Utilizzando un layer di tipo Layer.WFS funziona ma il metodo è deprecato e non viene gestita la style map.

Possibili soluzioni:

* Provare Mapserver version 6
* Provare Geoserver

# Proxy OL per layer vettoriali

Sviluppato proxy

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/services/proxy/proxy.ashx?url>=

<http://www.cartografiarl.regione.liguria.it/CartoWebNet2/services/proxy/proxy.ashx?url>

Impostata la stringa OpenLayers.ProxyHost = '…'; in Base.js

# Info e PopUp su livelli vettoriali

Sviluppata gestione info su livelli vettoriali in CallBacks.js (funzione onFeatureSelect)

Viene gestita la popUp in stile ballon che il link alla pagina di info

Due modalità

* dinamica
* statica

## Parametri

Vengono gestite le seguenti proprietà:

* infoPopUp: contiene il codice html da mostrare nel baloon.
* infoUrl: url della pagina da richiamare
* infoTarget: target della pagina
  + “\_blank” per aprire su una nuova finestra di browser
  + “panel” per aprire in un pannello ExtJS
* infoWidth: larghezza della finestra
* infoHeight: altezza della finestra

**Se impostato infoPopUp vengono ignorate le altre informazioni.**

Per la gestione dinamica degli attributi viene utilizzata la funzione OpenLayers.Style.createLiteral utilizzando la notazione ${nomeAttributo} dove nomeAttributo è il nome di una delle “properties” degli oggetti GeoJSON.

Es: “http://parodixp2.datasiel.net/pippo.html?id=${id}"

## Modalità statica

In questa modalità le informazioni sono definite a livello dei dati, cioè a livello del singolo oggetto GeoJSON

I parametri vanno definite nell’oggetto GeoJSON passato alla applicazione (vedi MappaCultura.html)

## Modalità dinamica

In questa modalità le informazioni sono definite nella configurazione del layer è quindi necessario configurare opportunamente il layer

I parametri vanno definiti nell’oggetto metadata della configurazione del layer:

# Installato OL 2.11

Installato OL versione 2.11 in E:\progetti\CartoWebNet2\lib\OpenLayers su parodis-dts-pc

# Immagini OL

Sostituito immagini

Openlayers\img\ cloud-popup-relative.png

OpenLayers\theme\default\img\close.gif

Messe anche in \CartoWebNet2\backup img openlayers

Quando si fa upgrade di OL ricordarsi si copiarle

# Impostato nuovo renderer SVG in OL

In Base.js ho impostato la seguente prorpietà:

OpenLayers.Layer.Vector.prototype.renderers = ["SVG2", “SVG”, "VML", "Canvas"];

# Test coordinate

Provato a caricare Punti Rete GNSS (Posizioni/1245.map livello L3254) su una mappa basata su google (stradario/satellite/ibrida) e confrontato con mappa del repertorio su ortofoto.

La sovrapposizione è quasi perfetta, comunque ampiamente accettabile, solo in alcune zone (montuose) le immagini satellitari Google non sono allineate (forse perché non ortorettificate)

**Test coordinate per livelli vettoriali**

Convertito punti GNSS con CartLab3 e sovrapposti al servizio WMS GB

* test servizio WFS MS
* test servizio GeoJSON con conversione dinamica

**TUTTI I TEST OK.**

# Installazione Newtonsoft.Json

Installato in UtilityBase per gestire il problema del CamelCase:

proprietà ASP.NET iniziano con maiuscola mentre js si aspetta le proprietà minuscole.

# Servizi

Solution “**services**” comprende i seguenti progetti:

**BLL**: Business Logic Layer, comprende le classi base.

**DAL**: Data Access Layer, comprende i metodi per l’accesso alla base dati.

**SrvConfig**: Comprende i servizi di configurazione.

P.e. getCongig.ashx, getLayers.ashx, ecc…

# NHIbernate

<http://www.theserverside.net/tt/articles/showarticle.tss?id=NHibernate>

<http://www.theserverside.net/tt/articles/showarticle.tss?id=NHibernateP2>

<http://www.codeproject.com/KB/architecture/NHibernateBestPractices.aspx>

Gestione Composite keys:

<http://devlicio.us/blogs/anne_epstein/archive/2009/11/20/nhibernate-and-composite-keys.aspx>

## Installazione

Scaricato file in E:\progetti\FluentNHibernate

Aggiunto reference a NHibernate.dll nei progetti

Creata “New Solution Folder” e aggiunto xsd:

* nhibernate-configuration.xsd
* nhibernate-mapping.xsd

Creato classi:

* NHibernateHelper.cs: gestione configurazione e sessioni
* NHibernateCurrentSessionWebModule.cs: httpmodule che istanzia la sessione all’avvio della richiesta e la chiude alla fine

Configurazione:

Modificato web.config per gestire httpmodule per NHB in system.web e system.webServer.

Creato file di mapping .hbm.xml per ogni classe persistente.

Create file di configurazione /services/srvConfig/bin/hibernate.cfg.xml

## Problemi con serializzazione JSON

<http://stackoverflow.com/questions/286721/json-net-and-nhibernate-lazy-loading-of-collections>

## Problema many-to-many con attribute

**es. BaseLayer – Application e BaseLayer – Carta**

Risolta costruendo vista

* Costruita vista CWN2\_V\_APPL\_BASE\_LAYERS
* Oggetto ApplicationBaseLayers costruito su vista
* Relazione on-to-many tra Application e ApplicationBaseLayers

Sarà necessaria anche la vista CWN2\_V\_GRA\_BASE\_LAYERS con relativa classe MapBaseLayers

Fare una classe astratta BaseLayers da cui derivano le due classi ApplicationBaseLayers e MapBaseLayers

ALTERNATIVA modellare 3 entità

* The student\_course table is there purely to record the association between the two entities. It is managed by hibernate, and can contain no other data. The sort of data you want to record needs to be modelled as another entity. Perhaps you could a **one-to-many** association between Course and StudentResult (which contains the grade, etc), and then a **many-to-one** association between StdentResult and Student.
* <http://stackoverflow.com/questions/1153409/manytomany-hibernate-question-can-add-extra-field/1168740#1168740>
* <http://stackoverflow.com/questions/1212058/how-to-make-a-composite-primary-key-java-persistence-annotation/1252317#1252317>

## Problema sicurezza IIS

<http://nhforge.org/wikis/howtonh/run-in-medium-trust.aspx>

<http://www.d80.co.uk/post/2011/03/21/Nhibernate-FileIOPermission-Exception-because-of-CodeAccessSecurity.aspx>

# Base dati metadati

### Applicazione

#### Tavola CWN2\_APPLICATIONS

### Opzioni Mappa

#### Tavola CWN2\_MAP\_OPTIONS

### Layout

#### Tavola CWN2\_LAYOUT

### Widget

#### Tavola CWN2\_WIDGET

#### Tavola di relazione CWN2\_APPL\_WIDGET

### Bottoni

#### Tavola CWN2\_TB\_ITEM

#### Tavola CWN2\_TB\_ITEM\_OPTIONS

Contiene una serie di opzioni relative al bottone. Ogni riga contiene due campi KEY e VALUE.

Viene utilizzata nella configurazione per valorizzare la hash-table:

Configuration.toolbar.itemGroups[n].items[m].options

Per il bottone loadLayers bisogna inserire le righe:

**"panels"**: **[**"canali"**]**,

**"canaliTematiciTreeServiceUrl"**: "/CartoWebNet2/services/loadLayerSrv/getCanaliTematiciTree.ashx?ID\_APPLICAZIONE=",

**"getLayersConfigServiceUrl"**: "/CartoWebNet2/services/loadLayerSvr/getLayersConfig.ashx?layers="

### Toolbar

#### Tavola CWN2\_TB\_TOOLBARS

#### Tavola CWN2\_TB\_ITEM\_GROUPS

#### Tavola di relazione CWN2\_TB\_REL\_TOOLBAR\_GROUP

#### Tavola di relazione CWN2\_TB\_REL\_GROUP\_ITEM

### Layer di Base

#### Tavola CWN2\_BASE\_LAYERS

#### Tavola di relazione CWN2\_APPL\_BASE\_LAYERS

Relazione tra CWN2\_APPLICAZIONE e CWN2\_BASE\_LAYERS

#### Tavola di relazione CWN2\_MAP\_ BASE\_LAYERS

Relazione tra SIT\_CATALOGO e CWN2\_BASE\_LAYERS

### Canali Tematici

#### Tavola CWN2\_CANALI

#### Tavola di relazione CWN2\_CANALI\_LIVELLI

#### Tavola di relazione CWN2\_APPL\_CANALI

### Gestione Layer

Gestire Ereditarietà: su unica tavola con attributo TYPE (es Layer 🡪 WMS,WFS,ecc.)

Mappatura Layer/CWR\_LIVELLI/CWN2\_V\_LAYERS

|  |  |  |  |
| --- | --- | --- | --- |
| Layer | CWR\_LIVELLI | CWN2\_V\_LAYERS | NOTE |
|  | COD\_CATALOGO | ID\_MAP | FK su CWN2\_V\_MAPS |
| Id | CODICE\_LIVELLO | ID | PK |
| Type | TYPE | TYPE | **NUOVO CAMPO** VARCHAR2(20)  (WMS/WFS/VECTOR) |
| Name | -- | NAME | CALCOLATO NELLA VISTA COME  L || TO\_CHAR(CODICE\_LIVELLO) |
| Projection | CRS | PROJECTION | ‘EPSG:”||TO\_CHAR(CRS) |
| Visible | VISIBLE | VISIBLE | **NUOVO CAMPO**  (INTEGER) – Impostare di default a 1 (true) |
| Attribution | ATTRIBUTION | ATTRIBUTION | **NUOVO CAMPO**  VARCHAR2(200) |
| MinScale | SCALA\_MIN | MAX\_SCALE |  |
| MaxScale | SCALA\_MAX | MIN\_SCALE |  |
| NOrder | ORDINE | N\_ORDER |  |
| Opacity | TRASPARENZA | OPACITY | CALCOLARE IN VISTA COME (100 – TRASPARENZA)/100 |
| WmsParams.Url e  WfsParams.Url | WMS\_PARAM\_URL | WMS\_PARAM\_URL | **NUOVO CAMPO**  VARCHAR2(200)  Valorizzato da procedura di creazione servizi WMS/WFS – Creare programma per valorizzazione pregresso oppure rigenerare i tutti i servizi |
| WmsParams.Name  E  WmsParams.TypeName | -- | WMS\_PARAM\_NAME | CALCOLATO NELLA VISTA COME  L || TO\_CHAR(CODICE\_LIVELLO) |
| WmsParams.Transparent | WMS\_PARAM\_TRANSPARENT | WMS\_PARAM\_TRANSPARENT | **NUOVO CAMPO** (INTEGER) – Impostare di default a 1 (true) |
| LoadVoid | LOAD\_VOID | LOAD\_VOID | **NUOVO CAMPO** (INTEGER) – Impostare di default a 0 (false) |
| LayerVectorProtocol.Type | PROTOCOL\_TYPE | PROTOCOL\_TYPE | **NUOVO CAMPO** (VARCHAR2) –  IMPOSTARE DI DEFAULT A “HTTP” |
| LayerVectorProtocol.Options.Format | PROTOCOL\_FORMAT | PROTOCOL\_FORMAT | **NUOVO CAMPO** (VARCHAR2) –  IMPOSTARE DI DEFAULT A “GeoJSON” |
| LayerVectorProtocol.Options.Format | PROTOCOL\_URL | PROTOCOL\_URL | **NUOVO CAMPO** (VARCHAR) – URL dipende da servizio implementato |
| LayerLegend.Label | DESC\_LIV | LEGEND\_LABEL |  |
| LayerLegend.Icon | LEGEND\_ICON | LEGEND\_ICON | **NUOVO CAMPO**  VARCHAR2(200)  **WMS**: Valorizzato da procedura di creazione servizi WMS/WFS – Creare programma per valorizzazione pregresso oppure rigenerare i tutti i servizi  **VETTORIALI**: deve essere valorizzato con la url della icona |
| LayerLegend. PopUpUrl | LEGENDA\_URL | LEGEND\_POPUP\_URL |  |
| LayerLegend. PopUpWidth | LEGEND\_POPUP\_WIDTH | LEGEND\_POPUP\_WIDTH | **NUOVO CAMPO** |
| LayerLegend. PopUpHeight | LEGEND\_POPUP\_HEIGHT | LEGENDA\_POPUP\_HEIGHT | **NUOVO CAMPO** |

**Modificata tavola CWR\_LIVELLI**

**SCRIPT:**

ALTER TABLE SIT.CWR\_LIVELLI

ADD (TYPE VARCHAR2(20))

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (VISIBLE INTEGER DEFAULT 1)

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (ATTRIBUTION VARCHAR2(200))

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (WMS\_PARAM\_URL VARCHAR2(200))

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (WMS\_PARAM\_TRANSPARENT INTEGER DEFAULT 1)

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (LOAD\_VOID INTEGER DEFAULT 0)

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (PROTOCOL\_TYPE VARCHAR2(20) DEFAULT 'HTTP')

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (PROTOCOL\_FORMAT VARCHAR2(20) DEFAULT 'GeoJSON')

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (PROTOCOL\_URL VARCHAR2(200))

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (LEGEND\_ICON VARCHAR2(200))

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (LEGEND\_POPUP\_WIDTH NUMBER)

/

ALTER TABLE SIT.CWR\_LIVELLI

ADD (LEGEND\_POPUP\_HEIGHT NUMBER)

/

Update CWR\_LIVELLI set TYPE = 'WMS';

Creata vista CWN2\_V\_MAPS

Creata vista CWN2\_V\_LAYERS

### Gestione Classi

Oggetto Class

Serve per costruzione legenda per livelli multiclasse e per gestione stili per livelli vettoriali multiclasse

|  |  |  |  |
| --- | --- | --- | --- |
| Class | CWR\_CLASSI | CWN2\_V\_CLASSES | NOTE |
| Id | -- | ID | Generato con rownum |
| IdLayer | CODICE\_LIVELLO | ID\_LAYER |  |
| NumClass | NUM\_CLA | NUM\_CLASS |  |
| Filter | WHERE\_CLA | FILTER |  |
| ClassLegend. Label | DESC\_CLA | LEGEND\_LABEL |  |
| ClassLegend.Icon | LEGEND\_ICON | LEGEND\_ICON | **NUOVO CAMPO**  VARCHAR2(200)  **WMS**: Valorizzato da procedura di creazione servizi WMS/WFS – Creare programma per valorizzazione pregresso oppure rigenerare i tutti i servizi  **VETTORIALI**: deve essere valorizzato con la url della icona |
| Lista StyleMapConfig |  |  | Oggetto StyleMap  Vedi VISTA CWN2\_V\_STYLEMAPS |

Modificata tavola CWR\_CLASSI

Creata Vista CWN2\_V\_CLASSES

### Gestione Stili Vettoriali

Classe StyleMap

Aggiornata la tavola SIT\_STILI con i nuovi campi

CAMPI:

**FILL\_OPACITY**: NUMBER (REAL)

* Opacità del riempimento dei poligoni
* Dominio: 0 – 1

**STROKE\_OPACITY**: NUMBER (REAL)

* Opacità del bordo dei poligoni e delle linee
* Dominio: 0 – 1

**STROKE\_LINE\_CAP**: VARCHAR2(20) **NON GESTITO**

* Tipo di estremità della linea
* Dominio
  + round (default)
  + butt
  + square

**STROKE\_DASH\_LINE**: VARCHAR2(20) **NON GESTITO**

* Tipo di linea
* Dominio
  + dot
  + dash
  + dashdot
  + longdash
  + longdashdot
  + solid (default)

**EXTERNAL\_GRAPHIC**: VARCHAR2(500)

* URL della immagine (png/gif/jpeg) da utilizzare

**GRAPHIC\_WIDTH**: NUMBER (INTEGER)

* Larghezza in pixel dell’immagine

**GRAPHIC\_HEIGHT**: NUMBER (INTEGER)

* Altezza in pixel dell’immagine

**GRAPHIC\_OPACITY**

* Opacità dell’immagine
* Dominio: 0 – 1

**GRAPHIC\_X\_OFFSET, GRAPHIC\_Y\_OFFSET**: NUMBER (INTEGER)

* Offset dell’immagine per la centratura (default punto mediano del lato superiore)

**CURSOR**: VARCHAR2(20) **NON GESTITO**

* Cursore sull’hover del mouse
* Dominio:
  + pointer (default)

**DISPLAY**: VARCHAR2(20) **NON GESTITO**

* Se impostato a “none” le feature non vengono visualizzate

**RENDER\_INTENT**: VARCHAR2(20)

* Situazione nella quale utilizzare lo stile.
* Dominio
  + default (stile normale)
  + hilite (utilizzato quando si fanno selezioni di features)

ALTER TABLE SIT.SIT\_STILI ADD

(FILL\_OPACITY NUMBER DEFAULT 1,

STROKE\_OPACITY NUMBER DEFAULT 1,

STROKE\_LINE\_CAP VARCHAR2(20 BYTE),

STROKE\_DASH\_LINE VARCHAR2(20 BYTE),

EXTERNAL\_GRAPHIC VARCHAR2(500 BYTE),

GRAPHIC\_WIDTH INTEGER,

GRAPHIC\_HEIGHT INTEGER,

GRAPHIC\_OPACITY NUMBER DEFAULT 1,

GRAPHIC\_X\_OFFSET INTEGER,

GRAPHIC\_Y\_OFFSET INTEGER,

CURSOR VARCHAR2(20 BYTE),

DISPLAY VARCHAR2(20 BYTE),

RENDER\_INTENT VARCHAR2(20 BYTE)

)

CAMPI ATTUALMENTE GESTITI

Creata Vista CWN2\_V\_STYLEMAPS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| StyleMap | SIT\_STILI | CWN2\_V\_CLASSES | LABEL SU FORM | NOTE |
| Id | CODICE\_STILE | ID |  |  |
| IdLayer | CODICE\_LIVELLO | ID\_LAYER |  |  |
| NumClass | NUM\_CLA | NUM\_CLASS |  |  |
| RenderIntent | RENDER\_INTENT | **RENDER\_INTENT** | Render (default/hilite) | NUOVO CAMPO |
| Style |  |  |  |  |
| fillColor | COLOR | FILL\_COLOR | Color |  |
| fillOpacity | FILL\_OPACITY | **FILL\_OPACITY** | Fill Opacity | NUOVO CAMPO |
| strokeColor | OUTLINECOLOR | STROKE\_COLOR | OutLine Color |  |
| strokeOpacity | STROKE\_OPACITY | **STROKE\_OPACITY** | Stroke Opacity | NUOVO CAMPO |
| strokeWidth | WIDTH | STROKE\_WIDTH | Width |  |
| strokeLinecap | STROKE\_LINE\_CAP | STROKE\_LINE\_CAP | Stroke Linecap | ATTUALMENTE NON GESTITO. MESSO DI DEFAULT A ‘round’ |
| strokeDashstyle | STROKE\_DASH\_LINE | STROKE\_DASH\_LINE | Stroke Dashstyle | ATTUALMENTE NON GESTITO. MESSO DI DEFAULT A ‘solid’ |
| pointRadius | STYLE\_SIZE | POINT\_RADIUS | Size |  |
| externalGraphic | EXTERNAL\_GRAPHIC | **EXTERNAL\_GRAPHIC** | External Graphic | NUOVO CAMPO |
| graphicWidth | GRAPHIC\_WIDTH | **GRAPHIC\_WIDTH** | Graphic Width | NUOVO CAMPO |
| graphicHeight | GRAPHIC\_HEIGHT | **GRAPHIC\_HEIGHT** | Graphic Height | NUOVO CAMPO |
| graphicOpacity | GRAPHIC\_OPACITY | **GRAPHIC\_OPACITY** | Graphic Opacity | NUOVO CAMPO |
| graphicXOffset | GRAPHIC\_X\_OFFSET | **GRAPHIC\_X\_OFFSET** | Graphic X Offset | NUOVO CAMPO |
| graphicYOffset | GRAPHIC\_Y\_OFFSET | **GRAPHIC\_Y\_OFFSET** | Graphic Y Offset | NUOVO CAMPO |
| rotation | ANGLE | ROTATION | Angle |  |
| graphicName | SYMBOL | GRAPHIC\_NAME | Symbol |  |
| cursor | CURSOR | CURSOR | Cursor | ATTUALMENTE NON GESTITO. MESSO DI DEFAULT A ‘pointer’ |
| display | DISPLAY | DISPLAY | Display? | ATTUALMENTE NON GESTITO. MESSO DI DEFAULT A null |
|  |  |  |  |  |

### Gestione Canali

Costruita vista CWN2\_V\_APPL\_CANALI

Costruita vista CWN2\_V\_CANALI\_LAYERS

### Aggiornamento \_CreaMapFileCS.asp per caricamento parametri

Procedure di popolamento campi url per layer e classi in gestione servizi (N:\CartoWebNET\\_CreaMapFileCS.asp)

Popolare i campi:

* CWR\_LIVELLI. WMS\_PARAM\_URL
* CWR\_LIVELLI. LEGEND\_ICON
  + se layer multiclasse usare icona “/CartoWebNet2/img/legend/classi.gif"
* CWR\_CLASSI. LEGEND\_ICON

Creata procedura UpdateParametriLivelli e UpdateParametriClassi

# Servizi di Configurazione

### Datasiel.Cwn2.Core

Class Library contenente le classi di base.

Utilizzo NHIbernate per la persistenza.

Utilizzo NewtonSoft Json per la serializzazione in JSON.

Classe Configuration contiene l’oggetto completo di configurazione.

### Metodo Configuration.getJsonConfig

Metodo statico che ritorna il json della configurazione

getJsonConfig(int idApp, int idMap, string listaLivelli)

idApp = id applicazione, se non valorizzato impostato a 0 (applicazione di default)

idMap = id mappa (codice\_catalogo), se non valorizzato impostato a 0 (nessuna mappa)

listaLivelli = lista degli id layer (CODICE\_LIVELLO)

Richiama il costruttore di Configuration che costruisce la configurazione.

Se idMap è valorizzato e ha dei BaseLayer associati carica solo i BaseLayer associati alla mappa altrimenti carica i BaseLayer associati alla applicazione

Se idMap è valorizzato carica i Layer associati alla mappa

Se listaLivelli è valorizzato carica i Layer con codice nella lista

### Metodo getJsonLayersConfig(string listaLivelli)

Metodo statico che ritorna il json della configurazione di una lista di layer.

### Metodo getJsonLayersConfig(int idMap)

Metodo statico che ritorna il json della configurazione dei layer associati ad una mappa.

# Visual Studio 2010

Installato VS 2010 (con framework 4.0)

Upgrade progetto a VS2010/FW4.0

Modificato pool applicazioni da DefaultAppPool a ASP.NET v4.0

Vedi <http://technet.microsoft.com/it-it/library/cc731755(WS.10).aspx>

# Gestione stylemap vettoriali

Ogni classe ha + stylemap (per ogni render intent) e un filtro.

Se layer ha una sola classe filter è null e viene ignorato e vengono applicate le stylemap della classe a tutto il layer

Nel LayerFactory vengono create le stylemap per il livello

I filtri per le classi (WHERE\_CLA) sono gestiti come filtri CQL (Attenzione non funziona l’operatore IN)

Vedi <http://openlayers.org/dev/examples/cql-format.html>

# jQuery

Utilizzo jQuery come libreria di base per ExtJs

<http://docs.jquery.com/Tutorials:Using_Ext_With_jQuery>

Installato in lib/jQuery, sia la varsione min che quella non min

# Test Configurazione layer Vector

Test costruzione layer di tipo VECTOR-GeoJSON

Definita carta codice\_catalogo 946 codice\_grafico 3853 codice\_tema 3037

Layer codice 2227 di tipo GeoJSON puntuale

Layer codice 2228 di tipo GeoJSON loadVoid

# Json.load

Rivisto con jQuery.

<http://api.jquery.com/jQuery.getJSON/>

# Caricamento Asincrono

Implementato il caricamento asincrono della configurazione e delle richieste json in generale. Il meccanismo richiede la definizione e il passaggio di funzioni di callback.

# Oggetto Application

Creato oggetto Application che gestisce la applicazione.

Creato metodo init per la inizializzazione della applicazione.

Il metodo richiama il caricamento della configurazione in modalità asincrona, la creazione della mappa OL e la costruzione del layout.

Viene richiamata una eventuale funzione di Callback alla fine della inizializzazione. I client che devono effettuare operazioni dopo la init devono definire una funzione di callback e passarla al metodo init.

# Oggetto LayerManager

Creato oggetto LayerManager che gestisce i layer per i pannelli accessori (legenda, remove layer, transparency).

Contiene uno store ExtJs contenente i layer visibili in legenda. Lo store viene inizializzato alla inizializzazione della applicazione Application.init()

Contiene i metodi per aggiungere e rimuovere layer dalla applicazione mantenendo sincronizzata mappa, configurazione e pannelli accessori

# Gestione flag debug

Debug è impostato di default è false.

Viene impostato a true se

* Viene passato parametro “debug=true” in querystring
* Viene impostata la variabile “CWN2.Application.debug = true” via javascript.

Impostazione in querystring: viene controllato in Application.init() e impostato se = true

Impostazione via javascript. Vedi esempio EsempioPanel.js

# Problema parsing json con lettere accentate

IE 8 non riesce a fare il parsing del json con lettere accentate.

E’ necessario mettere i file json su una applicazione IIS e impostare nel web.config l’encoding nel charset opportuno iso-8859-1

Ho spostato le configurazioni in services/Configuration/static e impostato la sezione del web.config

# Oggetto CWN2.Map

Definito oggetto CWN2.Map con costruttore.

Rappresenta l’interfaccia verso l’applicazione.

Implementato metodi base:

* costruttore
* load
* getUrlParam
* loadFeature
* hilite
* ….

# StyleMap di Default

Implementata gestione della StyleMap di default in layerFactory

Nella configurazione se non definisco stili sulla base dati la stylemap viene definita come array vuoto

"styleMaps": []

In layerFactory ho creato un array renderIntents contenente i render intent gestiti con attributi name e defaultStyle (attualmente copiati dei default di OL)

Creato metodo getDefaultStyleByIntent(renderIntent):

# Mapserver 6.0

Vedi test.map su N:\mapfiles\test

#### Migrazione

[http://mapserver.org/MIGRATION\_GUIDE.html #mapserver-5-6-to-6-0-migration](http://mapserver.org/MIGRATION_GUIDE.html#mapserver-5-6-to-6-0-migration)

Fatto test su reati.map

* **Gestione escape**:
  + Sostituire \” con \\” Es: SHAPEPATH "C:\ms4w\shapefiles\\"
* **Output format**:
  + sostituire GD/PNG e GD/JPEG con AGG/PNG e AGG/JPEG
* **Abilitazione servizi OGC** 
  + Aggiungere "ows\_enable\_request" "\*" in MAP::WEB::METADATA section
* **Expression parsing**: controllare (where\_cla?)
* **Label Style**: cambiata gestione box: levare BACKGROUNDCOLOR,BACKGROUNDSHADOWCOLOR, BACKGROUNDSHADOWSIZE. Per gestire box vedi documento. #
* **Line Styling**: vedi documento
* **Rotazione simboli**: All symbols now rotated anticlockwise following the ANGLE parameter
* **SYMBOLSET:** STYLE all’interno dei SYMBOL è stato rinominato PATTERN e spostato in STYLE.
  + Modificato examples.sym su dcarto3
  + Problema: come fare a convertire tutti i simboli vecchi?

**CASINO SULLE LINESTRING**

#### WFS

MapServer 5 ha problemi a gestire le richieste in POST (non viene mandata la geometria), quindi l'utilizzo di layers di tipo Openlayers.Vector con formato WFS non funziona. Vedi configurazione E:\progetti\CartoWebNet2\config\ProvaCaricamentoWFS1.json

Provo con MS 6:

Livello GNSS da shape file: OK

ATTENZIONE: Bisogna impostare versione WFS a 1.0.0 altrimenti coordinate invertite

Livello TUR\_STR\_ALB\_WGS84 da ORACLE

* Bisogna impostare la strategy BBOX
* non riesce a scrivere le geometrie: NON BISOGNA METTERE IN LAYER::METADATA
  + gml\_featureid "ID" ## REQUIRED
  + gml\_geometries "GDO\_GEOMETRY"

Livello OBIETTIVI da POSTGIS.

Effettua anche la conversion da GB a WGS84!!!!

Livello COMUNI\_5000 da POSTGIS: molto lento

#### GeoJSON

Implementato uscita GeoJSON in MS

**OGR OUTPUT**

<http://mapserver.org/output/ogr_output.html#ogr-output>

<http://osgeo-org.1803224.n2.nabble.com/OGR-Output-and-GeoJson-td6203849.html>

**Mapfile:**

Definire OUTPUTFORMAT

OUTPUTFORMAT

NAME "GeoJSON"

DRIVER "OGR/GeoJSON"

MIMETYPE "text/plain"

FORMATOPTION "STORAGE=stream"

FORMATOPTION "FORM=simple"

END

Nel LAYER::METADATA aggiungere

"wfs\_getfeature\_formatlist" "GeoJSON"

NB: attualmente imposto mimetype “text/plain” per potrerlo leggere direttamente dal browser ma dovrebbe essere “application/json”.

Se si vuole usare anche nella WMS getFratureInfo bisogna

* Impostare MMEMETYPE a “application/json”
* Aggiungere "wms\_feature\_info\_mime\_type" "text/csv" nel LAYER::METADATA

**Url dei servizi è del tipo**

http://dcarto3.datasiel.net/MapServer/6.0/mapserv.exe?MAP=E:\\progetti\\mapfiles\\test\\test.map&SERVICE=WFS&VERSION=1.0.0&REQUEST=getfeature&TYPENAME=ALBERGHI&BBOX=4.7458076860073,42.20866415158,12.798786200511,46.223831521126&OUTPUTFORMAT=GeoJSON

NB: In CWN2.layerFactory ho dovuto impostare l’opzione ignoreExtraDims = true perchè MS crea il GeoJSON 3d (x,y,z) e OL non lo legge

**Si potrebbe implementare anche con metodo TEMPLATE**

<http://www.osgeo.org/pipermail/mapserver-users/2009-July/062005.html>

<http://osgeo-org.1803224.n2.nabble.com/GeoJSON-output-from-MapServer-td3188987.html>

esempio mapfile <http://svn.osgeo.org/mapserver/trunk/msautotest/wxs/wfs_ogr.map>

Esempio template <http://svn.osgeo.org/mapserver/trunk/msautotest/misc/rfc36.tmpl>

# Livelli Base WMS

Implementato gestione di baseLayer WMS

Vedi mappa reati

Settato parametri per zoomLevels vedi: <http://trac.osgeo.org/openlayers/wiki/SettingZoomLevels>

# Gestione dinamica ZoomLevels per I layer di base

<http://trac.osgeo.org/openlayers/wiki/SettingZoomLevels>

In configurazione

mapOptions:

* eliminato minScale

configurazione layer Google:

* levato parametro minZoomLevel (calcolato dinamicamente)
* levato parametro maxZoomLevel (calcolato in base a maxScale)

**In layerFactory**

Impostato resolutions in costruttore per layer Google e OSM

**In mapManger**

**Registrato evento “move” sulla mappa che richiama la funzione onMove, se le dimensioni della mappa sono variate calcolo il nuovo minZoomLevel per i layer di base (implementato per Google) e l’array resolutions per i layer OSW e WMS**

# Gestione fractionalZoom

Se il baseLayer è WMS si potrebbe impostare il fractionalZoom a true.

Cosa succede se ho baselayer misti (wms e google)? Succede un casino.

**Si può settare solo se TUTTI i base layer sono WMS**

Nuova versione OL gestisce il client zoom con fractionalZoom anche per i cached layer:

<http://www.camptocamp.com/en/blog/2011/10/openlayers-client-zoom-feature/>

<http://elemoine.github.com/openlayers/examples/clientzoom.html>

Scaricato OL da GitHub in E:\progetti\CartoWebNet2\lib\openlayers-e960c19

Non sembra funzionare per layer Google.

Abilitato fractionalZoom solo se NON ci sono baseLayers Google

# Gestione initialExtent

In CWN2.configuration funzione \_loadConfig:

* Se impostato in QueryString parametro initialExtent e initialExtentCRS: viene calcolato initialExtent (con eventuale conversione di CRS) e impostato configuration.mapOptions.initialExtent
* Se sono impostati i parametri config.map.extent (letti da SIT\_GRAFICO) viene calcolato initialExtent e impostato.
* Altrimenti viene impostato pari a maxExtent

# Gestione noBaseControls

Gestione inibizione del caricamento dei controlli di navigazione definito flag **mapOptions.noBaseControls** in configurazione che se impostato a true non carica i controlli base.

# **Problema BBOX errato in request livello vettoriale**

A volte viene calcolato un extent errato e fallisce la

getFeature. Succede in particolare sui layer di base google.

OL quando inizializza la mappa senza div nel costruttore

crea una mappa di 1\*1 px che viene poi ridimensionata quando viene

fatto il render sul div.

All'atto del ridimensionamento viene ricalcolato il bound e impostato

con un extent errato.

Risolto con override del metodo OpenLayers.Strategy.BBOX.calculateBounds in OpenLayersExt.js

# **featureMngr**

Creato oggetto CWN2.featureMngr, per gestire le feature vettoriali

* Attivazione controlli di select, hilite, hover
* Gestione caricamento features
* Gestione callback sulla select
* Gestione hilite features

# **Custom scripts**

Creata dir **“\CartoWebNet2\lib\Cwn2\custom**” per le customizzazioni. In questa dir vanno messi gli script js che implementano funzionalità per particolari applicazioni (p.es: vedi infrastrutture).

Gli script di questa dir non vengono compilati nella libreria cwn2 ma devono essere inclusi dalla pagina chiamante.

# Organizzazione pagine/configurazioni

Creata dir \CartowebNet2\pages che contiene le pagine html suddivise in

* apps: pagine di test per applicazioni rilasciate
* test: Pagine che servono per testare funzionalità di interfaccia per cui è richiesta la verifica a video
* esempi
* prove: prove varie

Anche configurazioni sono organizzate nello stesso modo sotto /services/configuration/static

Creata pagina indice pages\index.html

# **Funzione selectFeature con solo hilite**

Implementata funzione che simula programmaticamente l’azione di hover sulla feature.

Evidenzia la feature e non fa la select.

Modificata featureMngr.selectFeature con option hiliteOnly: se true non viene fatta select ma highlight.

Create la funzione unhiliteFeature che richiama la funzione featureMngr.unselectFeature che fa unselect con hiliteOnly=true

# **Same Origin Policy e Proxy**

Requisito per utilizzo di cwn2 da applicazioni/pagine che non sono sullo stesso host della libreria cwn2 e che devono accedere a dati remoti è l’esistenza di un server proxy sullo stesso host che ospita le pagine. Questo a causa delle regole di sicurezza di “same origin policy” di javascript.

L’indirizzo del proxy deve essere poi impostato dalla pagina chiamante nella configurazione della map.init

var initConfig = {

appConfig: 'http://config.json',

divID: "cwn2\_map",

callBack: loadFeatures,

debug:true,

**proxy: “http://proxy.url”**

};

Il proxy così impostato viene poi utilizzato da cwn2 per tutte le request

* loadJson
  + caricamento configurazione
* proxy usato da OL
* servizi google (TODO)

Se non è impostato non viene utilizzato nessun proxy nella loadJson e come proxy OpenLayers viene utilizzato quello di default ('/CartoWebNet2/services/proxy/proxy.ashx?url='). Questo funziona solo su pagine che risiedono sullo stesso host della libreria cwn2.

Siti che non sono sullo stesso host delle librerie cwn2 DEVONO impostare il proxy.

Se non è disponibile un proxy:

* La configurazione di cwn2 deve risiedere sullo stesso host che contiene la pagina
* Eventuali dati vettoriali devono risiedere sulla stessa macchina
* Non è possibile accedere a servizi vettoriali (WFS o JSON) esterni
* Non è possibile accedere ai servizi google (geocoding e routing)

# **Gestione sfondi OSM/bianco con sfondi Google**

Se ridimensiono pagina su una mappa con sfondi google e sfondi di altro tipo quando ridimensiono la mappa lo sfondo google non si aggiorna (rimane disallineato)

Se ridimensionamento provoca solo un pan della mappa 🡪 disallineamento

Se invece provoca uno zoom (cambiamento di ZoomLevel) 🡪 OK

Se faccio pan dalla mappa 🡪 OK

**REGOLA: non si può usare sfondo bianco o sfondi OSM con layer google se la mappa è ridimensionabile**

# **Gestione stile popup default**

Creato stili in \stili\default\base.css

# Gestione Exception

Diversi livelli di exception

* 0: nessun messaggio, solo log
* 1: errore non bloccante
  + Mandare messaggio all’utente
  + Mandare log
* 2: errore bloccante
  + Mandare messaggio all’utente
  + Mandare log
  + Interrompere elaborazione

Exception con

* Level (0,1,2)
* Name
* Message

Creato Oggetto CWN2.util.handleException che in base al livello della exception:

* manda alert (messageBox)
* scrive log su console

# Test Suite

### Unit Test

Implementata UNIT TEST

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/lib/Cwn2/test/CWN2_UNIT_TEST.html>

Contiene gli unit test della maggior parte delle funzoini

### System Test

Implementata SYSTEM TEST (end-to-end) con la maggior parte delle funzionalità.

Basato su layout di tipo viewport

<http://parodis-dts-pc.datasiel.net/CartoWebNet2/lib/Cwn2/test/CWN2_UNIT_TEST.html>

# Deploy libreria JS e pagine

Configurato deploy con WebStorm

<http://www.jetbrains.com/webstorm/webhelp/remote-hosts.html>

Deploy di

* Libreria CWN2: \CartoWebNet2\lib\cwn2\src\\*.js (ESCLUSO SOTTODIR)
* Immagini: \CartoWebNet2\img (TUTTA)
* Pages: \CartoWebNet2\pages (TUTTA)
* Stili: \CartoWebNet2\stili (TUTTA)

# Calcolo percorsi con google

### Descrizione

Funzione attivabile attraverso un bottone.

La selezione del bottone determina l’apertura di un pannello accessorio che permette di definire un percorso da un punto di partenza ad un punto di arrivo utilizzando i servizi Google.

I punti di partenza ed arrivo possono essere definiti

* mediante un indirizzo
  + per comporre dinamicamente l’indirizzo vengono utilizzati i servizi di geocoding
* mediante la selezione di un punto sulla mappa
* mediante la posizione geografica dell’utente (se supportato dal browser)
* essere preventivamente impostati a partire da un comando esterno (attivabile per esempio dalla select di una feature o dalla popUp di una feature).

L’utente può definire il tipo di percorso

* auto
* piedi

L’utente può invertire il percorso

Il bottone “Calcola” effettua il calcolo e presenta il risultato.

Il risultato viene rappresentato cartograficamente sulla mappa e alfanumericamente (come diario di bordo) sul pannello accessorio.

Sul pannello accessorio vengono presentati:

* Informazioni generali sul percorso con indicazione della distanza e del tempo previsto
* Il diario di bordo del percorso con
  + Una riga per ogni tappa
  + Selezionando la riga si centra la mappa sulla tappa mostrando un icona con un tooltip contenente la descrizione della tappa

### Documentazione

Vedi: <http://javascript.html.it/articoli/leggi/3242/google-maps-v3-calcolare-i-percorsi-con-directions/5/>

API: <http://code.google.com/intl/it-IT/apis/maps/documentation/directions/>

API v3: <http://code.google.com/intl/it-IT/apis/maps/documentation/javascript/services.html#Directions>

Esempio: <http://lists.osgeo.org/pipermail/openlayers-dev/2009-October/005333.html>

### Interfaccia

La finestra di interfaccia è suddivisa in due parti: una superiore con parametri di input e parte inferiore con output (diario del viaggio)

Potrebbe essere sviluppata in due modi:

* Finestra popup flottante
* Panel accessorio in comune con la legenda (a fisarmonica)

### Oggetti Google

##### **DirectionRequest**

{

origin: LatLng | String,

destination: LatLng | String,

travelMode: TravelMode,

unitSystem: UnitSystem,

provideRouteAlternatives: Boolean,

region: String

}

Dove:

* origin e destination possono essere un oggetto LatLon o una String
* TravelMode può essere
  + TravelMode.DRIVING
  + TravelMode.WALKING
* UnitSystem = UnitSystem.METRIC
* provideRouteAlternatives = false
* region = “it”

##### **DirectionsResult**

Risposta del servizio è uno status code e un oggetto DirectionResult contenente

* **routes**[]: un array di **DirectionsRoute** con una sola occorrenza (non gestisco percorsi alternativi)

##### **DirectionsRoute**

* **legs**[]: un array di **DirectionsLeg** con una sola occorrenza (non imposto ViewPoint)
* **overview\_path:** array di **LatLon** (versione approssimata del percorso)
* **bounds**: contiene un **LatLngBounds**
* **copyrights**: testo con copyright da visualizzare sulla mappa

##### **DirectionsLeg**

* **steps**[]: array di **DirectionStep**
* **distance:** distanza sotto forma di oggetto **Distance**
* **duration:** oggetto **Duration**
* **start\_location:** oggetto **LatLon** di partenza
* **end\_location:** oggetto **LatLon** di arrivo
* **start\_address:** indirizzo di partenza (testo)
* **end\_address:** indirizzo di arrivo (testo)

##### **DirectionStep**

* **instructions:** contains instructions for this step within a text string
* **distance:** contains the distance covered by this step until the next step, as a **Distance** object
* **duration** contains the typical time required to perform the step as a **Duration** object
* **start\_location:** oggetto **LatLon** di partenza
* **end\_location:** oggetto **LatLon** di arrivo

##### **LatLng**

* lat
* lon

<http://code.google.com/intl/it-IT/apis/maps/documentation/javascript/reference.html#LatLng>

##### **LatLngBounds**

* LatLng
* LatLng

<http://code.google.com/intl/it-IT/apis/maps/documentation/javascript/reference.html#LatLngBounds>

##### **Distance**

* value: distanza in metri
* text: rappresentazione stringa della distanza

##### **Duration**

* value: durata in secondi
* text: rappresentazione stringa della durata

### Oggetti CWN2

#### googleGeocoder / geocodeHelper

StereoType: service provider / helper

Contiene metodi utili per il geocoding in particolare

* getAddressCombo(nomeCombo): costruisce una combo Ext per la ricerca indirizzi
* getAddressFromLatLng(lat,lon): effettua un reverse geocoding a partire da un punto
* getLatLonFromAddress(address): ritorna le coordinate di un indirizzo

in bottone find utilizzare metodo getAddressCombo per costruire la combo

In map interfacciare il metodo getAddressFromLatLng

#### routingMngr

StereoType: Information Holder / Service Provider

Gestisce la memorizzazione dei parametri necessari ad interrogare il servizio di routing google.

Permette di interrogare il servizio.

Gestisce la memorizzazione della risposta del servizio.

##### Proprietà:

* **origin**: contiene stringa con indirizzo di partenza e coordinate lat,lon
* **destination**: stringa con indirizzo di arrivo oppure e coordinate lat,lon
* **travelMode**: vedi google TravelMode
* **result**: risultato del calcolo
  + **distance**: oggetto Distance
  + **duration**: oggetto Duration
  + **steps**: array di DirectionsStep

##### Metodi:

###### **setOrigin**(address,lat,lon)

Imposta oggetto **origin** con address o con LatLng

###### **setDestination**(address,lat,lon)

Imposta oggetto **destination** con address o con LatLng

###### **getOrigin**()

Ritorna oggetto **origin**

###### **getDestination**()

Ritorna oggetto **destination**

###### **setTravelMode**(travelMode)

Imposta oggetto **travelMode**

###### **swapDirection**(): inverte origine e destinazione

###### **getRequestParams**():

Ritorna un oggetto DirectionsRequest con i parametri per effettuare la richiesta

Se origin.address è impostato usa address altrimenti crea oggetto LatLng con lat e lon

var origin = new google.maps.LatLng(lat, lon);

Lo stesso per destination

###### **calculate**(requestConfig): dove requestConfig è un oggetto di tipo DirectionsRequest

Effettua il calcolo del percorso.

* Richiede requestConfig mediante getRequestParams()
* Controlla che requestConfig sia corretta
  + origin e destination impostati
* Se requestConfig non è OK messaggio all’utente, altrimenti
* Interroga il servizio google alla risposta richiama processResult(directionsResult)
* Imposta oggetto **result**

###### **processResult**(directionsResult): dove directionsResult è un oggetto di tipo DirectionsResult

Gestisce la visualizzazione del risultato su mappa e su finestra di routing

* Imposta oggetto **result**
* Disegna il percorso sulla mappa
  + Scrive il copyright sulla mappa
  + Disegna tappe
  + Segnapunto partenza e arrivo con stile diverso
* Compone la sezione di output con:
* Intestazione con i dati del percorso:
  + Descrizione
  + Distanza
  + Durata
* Diario di bordo con elenco delle tappe result.routes[0].legs[0].steps[i]
  + Istruzioni (steps[i].instructions)
  + Distanza (steps[i].distance.text)
  + Durata (steps[i].duration.text)
  + Hover sulla riga
    - si evidenzia la tappa
    - si mostra un icona con un tooltip contenente la descrizione della tappa (si utilizza steps[i].start\_location) ??????
  + Selezionando la riga
    - si evidenzia la tappa
    - si mostra un icona con un tooltip contenente la descrizione della tappa
    - si centra la mappa (con zoom?)

### Oggetti interfaccia

##### **originAddressCombo**

Permette di indicare l’indirizzo di partenza.

Sull’evento select richiama setOrigin passando Address ,lat e lon

##### **destinationAddressCombo**

Permette di indicare l’indirizzo di arrivo.

Sull’evento select richiama setDestination passando Address, lat e lon

##### **originSelectOnMapBtn**

Permette di selezionare il punto di origine dalla mappa.

Prerequisito: controllo OL sulla mappa con un listener che al click sulla mappa

* richiama router.getAddressFromLatLng(lat,lon) per avere address
* richiama router**.**setOrigin(address,lat,lon) (\*)
* imposta originAddressCombo con address

oppure:

* richiama direttamente setOrigin(null,lat,lon)
* imposta originAddressCombo con una stringa fissa (es: “coordinate ….”)

##### **destinationSelectOnMapBtn**

Permette di selezionare il punto di destinazione dalla mappa.

* vedi originSelectOnMapBtn

##### **travelModeCombo**

Permette di impostare il travelMode.

Sulla onSelect richiama router.setTravelMode(travelMode)

##### **swapDirectionBtn**

Permette di invertire origine e destinazione.

Sulla selezione:

* richiama router.swapDirection()
* imposta il valore di originAddressCombo con quello di destinationAddressCombo
* imposta il valore di destinationAddressCombo con quello di originAddressCombo

##### **submitBtn**

Permette di effettuare il calcolo.

Se origine e destinazione non sono impostati viene mandato messaggio all’utente, altrimenti:

* richiama getDirectionsRequest() per avere i parametri della request
* chiama il servizio google passando il DirectionsRequest e una funzione di callback processResults

Creato controllo generico **CWN2.controls.getMapCoordinatesOnClick** che dopo click sulla mappa richiama una funzione di callback passando le coordinate (mappa, display, wgs84) eventuali parametri da passare alla callback devono essere indicati nella activate

### Impostazione Lingua

**NOTE BENE**: per impostare la lingua in cu isono espresse le indicazioni ritornate dal servizio bisogna opportunamente impostare il paramerto language nella queryString della include della libreria Google

Es:

<script src="http://maps.google.com/maps/api/js?sensor=false&v=3.2&language=it"></script>

<script src="http://maps.google.com/maps/api/js?sensor=false&v=3.2&language=en"></script>

# Gestione richiamo funzioni di callback sugli eventi sulle feature

I layer possono registrare delle azioni da eseguire sugli eventi

* onFeatureSelect
* onFeatureUnselect
* onFeatureOver
* onFeatureOut

Le azioni sono implementate da apposite funzioni di callback che vengono richiamate sull’evento. Alle funzioni viene passato l’oggetto feature su cui è avvenuto l’evento.

La registrazione viene gestita dal featureMngr mediante il metodo registerCallback.

**Meccanismo di registrazione**

featureMngr mantiene un oggetto (**registeredCallbacks**) contenente gli array delle funzioni di callback registrate per ogni evento

Il metodo featureMngr. **registerCallback** prende in input:

* evento (onFeatureOver, ecc…..)
* funzione di callback

e aggiunge la funzione all’array delle funzioni per quell’evento.

La funzione richiamata sull’evento (es: onFeatureOver) cerca nel dizionario le funzioni con chiave relativa “onFeatureOver” e le esegue.

La ricerca viene fatta dalla funzione **getRegisteredCallbacks** che prende in input la chiave (es: “onFeatureOver”) e ritorna un array di funzioni

**Per configurazione una azione su un layer:**

1. Prerequisito: una funzione myFunction() che effettua l’azione richiesta.
2. In configurazione esiste l’array **actions** contentente un dizionario con chiave=**event**, value =**callback**:
   * event deve essere impostato con l’evento desiderato (es: onFeatureOver)
   * callback deve essere impostato con il nome della funzione da richiamare (myFunction)

esempio:

“actions”: [

{

“event”: “onFeatureOver”,

“callback” : “myFuncion”

}

]

layerMngr.init guarda nelle action dei layer e registra le funzioni mediante il metodo featureMngr. **registerCallback**

Eventi gestiti:

onFeatureOver

onFeatureOut

onFeatureSelect

onFeatureUnselect

**Per configurare una azione su un layer:**

Prerequisito: una funzione cWN2.fatureMngrActions.myFunction() che effettua l’azione richiesta.

In configurazione esiste l’array **actions** contentente un dizionario con chiave=**event**, value =**callback**:

* event deve essere impostato con l’evento desiderato (es: onFeatureOver)
* callback deve essere impostato con il nome della funzione da richiamare (myFunction)

esempio:

“actions”: [

{

“event”: “onFeatureOver”,

“callback” : “myFuncion”

}

]

**BASE DATI**

Creata tavola

CWN2\_LAYER\_ACTIONS

* ID
* ID\_LAYER
* EVENT
* CALLBACK

**SERVIZI**

Nuova classe LayerAction che punta alla tavola CWN2\_LAYER\_ACTIONS

Aggiungere bag “Actions” a layer.hbm.xml

Aggiungere IList<Action> Actions a classe Layer

# Installazione librerie JS su Dcarto3

Ho installato le librerie js (cartowebnet/lib/\*) su **dcarto3**

Creato procedura deployment di lib/cwn2 con WebStorm

## Multilinguismo

REQUISITO:

Implementare un sistema che permette di impostare le etichette della interfaccia in varie lingue e di definire la lingua di interesse come parametro di ingresso (parametro di configurazione o nella querystring)

Definito oggetto CWN2.i18n che contiene:

variabile “language” con la lingua impostata (default “it”)

oggetto “dict” contenente i testi nelle varie lingue

funzione get(string) che ritorna la stringa del dizionario nella lingua impostata

Impostazione della lingua:

* Default è “it” (italiano)
* Impostabile in configurazione layout (proprietà “language”)
* Nelle opzioni di init della applicazione (proprietà “language”)

# CouchDB

**Utilizzare couchDB per configurazioni**

Guida: <http://guide.couchdb.org/>

import file: <http://stackoverflow.com/questions/790757/import-json-file-to-couch-db>

jQuery <http://daleharvey.github.com/jquery.couch.js-docs/symbols/index.html>

Installato su **DCARTO3** come servizio in C:\Program Files\Apache Software Foundation\CouchDB

Modificato “C:\Program Files\Apache Software Foundation\CouchDB\etc\couchdb\**local.ini**”

* **bind\_address = 10.10.3.97**
* **allow\_jsonp = true**

Spostato database su E:\Progetti\CartoWebNet2\services\Configuration\static\couchdb e modificato “C:\Program Files\Apache Software Foundation\CouchDB\etc\couchdb\**default.ini**”:

* **database\_dir = E:\Progetti\CartoWebNet2\services\Configuration\static\couchdb**
* **view\_index\_dir = E:\Progetti\CartoWebNet2\services\Configuration\static\couchdb**

Creato DB “cwn2\_config”

Aggiunta configurazioni di prova

Test Caricamento configurazione da cwn2 con JSONP: **OK!**

Accesso a Futon: <http://dcarto3.datasiel.net:5984/_utils/>

Per accedere ad una configurazione

* <http://dcarto3.datasiel.net:5984/cwn2_config/prova>

Installato su **srvcarto.regione.liguria.it** (aperto porta 5984 sul firewall)

# Refactoring oggetti

**Effettuato refactoring con creazione oggetti costruttore**

* **Application**
* **Map**
* **Layout**
* **FeatureManager**

**Gestione multi-mappa**

# Installato OL 2.12

Installato OL versione 2.12 in E:\progetti\CartoWebNet2\lib\OpenLayers su parodis-dts-pc

# LayerSwitcher2

Creato controllo CWN2.Control.LayerSwitcher2

# Zoom

Creato controllo CWN2.Control.Zoom

# Build librerie JS

## Proj4js:

* compressione libreria proj4js.js
* unione proj4js e defs

Includere nelle pagine proj4js-debug o proj4js-min

Script ANT in /cartowebnet2/lib/proj4js/build.bat costruisce librerie

## OpenLayers:

Per eseguire la build di Openlayers è stata predisposta la procedura:

**/CartoWebNet2/lib/OL/build/build.bat**

Questa procedura utilizza il file di configurazione /CartoWebNet2/lib/OL/build/**custom.cfg** nel quale sono definiti i moduli OL da includere nella build.

La procedura prepara il file /CartoWebNet2/lib/OL/OpenLayers.js nelle versioni di sviluppo/test ed esercizio (versione minificata) e copia le relative versioni sui server di test (DCARTO3) ed esercizio (SRVCARTO).

# Funzione CWN2.Util.geoCode

Sviluppata funzione che effettua geocoding di indirizzo utilizzando servizi Google

Input: stringa contenente indirizzo

Output: oggetto GeoJSON puntuale

# Funzione FeatureLoader.loadFeatureByAddress

Sviluppata funzione che

* prende in input indirizzo
* effettua geocoding con google
* aggiunge oggetto vettoriale al layer

Input:

* indirizzo (stringa)
* livello
* html (html che compare nella popup del marker)
* zoom (livello)

Procedura:

* interrogazione servizio google
* cattura coordinate
* creazione oggetto vettoriale
* caricamento sul layer (con pulizia)
* centratura e zoom sulla mappa

# Gestione livelli TMS

Implementata gestione livelli TMS

**Attributi del livello di tipo TMS:**

* **name**: nome interno del layer
* **legend.label**: etichetta in legenda
* **url**: url del servizio TMS. (es: <http://parodis-dts-pc.datasiel.net/sfondi/ortofoto2007/>)
* **bounds**: estensione massima in EPSG:900903 (es: "815017,5412930,1132689,5590264")
* **minZoomLevel**: livello minimo di zoom a cui è visibile (es: 7)
* **maxZoomLevel**: livello massimo di zoom a cui è visibile (es: 20)

## Per creare sfondi TMS

Creare con programma **MapTiler**.

Installato mapTiler in C:\Program Files\MapTiler di parodis-dts-pc

**Trasformazione coordinate da GB a 900913 ha uno shift evidente, bisogna convertire dati in UTM/WGS84**

**Ottimizzazione png:**

<http://blog.klokantech.com/2008/11/png-palette-with-variable-alpha-small.html>

Utilizzare pngnq e pngcrush

* <http://pngnq.sourceforge.net/>
* Download: <https://sites.google.com/site/bitdepthtools/downloads>
* <http://pmt.sourceforge.net/pngcrush/>
* Download <http://sourceforge.net/projects/pmt/files/pngcrush-executables/1.7.15/>

Installati su parodixp2 e parodis-dts-pc in c:\programmi\pngnq\

Predisposto script python **pngCompress.py** per compressione prende in input la directory da comprimere(es: E:\progetti\sfondi\)

Progetto APTANA in C:\Users\parodi\My Documents\Aptana Studio 3 Workspace\pngCompress

Copiato sotto tools di CartoWebNet2/tools/pngCompress insieme agli eseguibili pngnq e pngcrush

Utilizzo: “pngCompress.py E:\progetti\sfondi\ortofoto2007”

**Prove con Ortofoto 2007**

* Tempi di elaborazione molto lunghi 5 giorni per MapTiler e 7 giorni per compressione png
* Dimensione dir 33GB (senza ottimizzazione png) e 15Gb con compressione

**Procedura:**

* Convertire file di partenza nel sistema UTL/WGS84
* Generare piramide con MapTiles, indicando come sistema di riferimento UTM zona 32 Nord / WGS84:

PROJCS["WGS 84 / UTM zone 32N",GEOGCS["WGS 84",DATUM["WGS\_1984",SPHEROID["WGS 84",6378137,298.257223563,AUTHORITY["EPSG","7030"]],AUTHORITY["EPSG","6326"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4326"]],UNIT["metre",1,AUTHORITY["EPSG","9001"]],PROJECTION["Transverse\_Mercator"],PARAMETER["latitude\_of\_origin",0],PARAMETER["central\_meridian",9],PARAMETER["scale\_factor",0.9996],PARAMETER["false\_easting",500000],PARAMETER["false\_northing",0],AUTHORITY["EPSG","32632"],AXIS["Easting",EAST],AXIS["Northing",NORTH]]

* Comprimere file con pnfCompress.py:
  + cd E:\progetti\CartoWebNet2\tools\pngCompress
  + pngCompress.py E:\progetti\sfondi\ortofoto2007
* Copiare directory su /progetti/sfondi/ di srvcarto

# Gestione restrictedExtent

impedire zoomOut oltre restricted extent

Gestito in evento moveend della mappa

Se il livello di zoom della mappa dopo la move è minore del livello di zoom assiciato al restrictedExtent faccio lo zoom sul restrictedExtent

# WMSGetFeatureInfo

Sviluppato bottone **infowms** che registra il controllo OpenLayers.Control.WMSGetFeatureInfo

<http://dev.openlayers.org/releases/OpenLayers-2.9.1/doc/apidocs/files/OpenLayers/Control/WMSGetFeatureInfo-js.html>

Esempi:

<view-source:http://dev.openlayers.org/releases/OpenLayers-2.11/examples/getfeatureinfo-popup.html>

Esempio che fa anche hilite

<view-source:http://dev.openlayers.org/releases/OpenLayers-2.11/examples/getfeatureinfo-control.html>

Infowms fa una **drillDown** su tutti i livelli WMS della mappa, se è impostato un **elenco di livelli nelle options del bottone** (es: layers : “REATI,RETE\_VIARIA”) fa la query SOLO dei layer nella lista, altrimenti tutti i layer WMS.

Vengono interrogati SOLO i layer **VISIBILI** in legenda

Il formato gestito (infoFormat) è di tipo 'application/vnd.ogc.gml'

**Lista di output**:

Pannello ExtJS contenente grid con la lista delle feature trovate ordinate per livello.

Viene visualizzata: label livello e label etichetta. La label etichetta viene configurata nelle **infoOptions.labelField**, se non presente viene messo **infoOptions.idField**, se non presente viene messo il primo attributo della feature.

Mettere in configurazione layer infoOptions attributi:

* labelField
* idField

Al click sulla riga viene aperto un pannello con scheda e hilite/zoom della feature

**Scheda:**

**Caso 1: infoUrl non xsl**

Se nella configurazione del layer è impostata una **infoOptions.infoUrl** e la url **non punta ad un file xsl:** viene costruita una MediaWindow con la pagina ritornata dalla url le variabili (es: id) vengono gestite nel solito modo (es: http://pippo/pluto.asp?id=${gid}).

**Caso 2: infoUrl xsl**

Se è impostata una **infoUrl** che punta ad un file **.xsl** nella configurazione del layer, applicare xslt a doc e scrivere output su finestra extjs. **TODO (vedi** <http://dev.abiss.gr/sarissa/index.html>)

**Caso 3: infoUrl non impostata, fieldMapping impostato**

Se non è impostata infoUrl ed è impostata una **infoOptions.fieldMapping** nella configurazione del layer: costruire una tabella (gridPanel) con la label dei campi e il valore.

**Caso 4: infoUrl non impostata, fieldMapping non impostato**

Se non è impostata né infoUrl né fieldMapping: costruire una tabella contenente tutti gli attributi del layer con nome campo e valore.

**Hilite feature:**

Utilizzato layer di appoggio vettoriale (1 per ogni layer di base con nome = <nome\_layer\_base>\_INFO) e rendering della geometria ritornata dalla featureInfo (se esiste)

Stile di rendering:

* definito nelle styleMap della prima classe del layer WMS con renderIntent “hilite”
* oppure default definito nel codice del bottone “infowms“

**--- --- --- ---**

**NOTE:**

**Per avere geometria in getFeatureInfo bisogna aggiungere i seguenti metadati nel MapFile:**

"GML\_GEOMETRY\_TYPE" "point" (tipo di geometria)

"GML\_GEOMETRIES" "geometry" (nome della geometria nel gml)

**ATTENZIONE** impostare il tipo di geometria a multipolygon e multiline per poligoni e linee

**---**

**Gestione configurazione:**

Mettere in configurazione layer infoOptions attributi:

* labelField: nome campo da utilizzare per la lista delle feature
* idField: nome del campo id della feature
* infoUrl: url da richimare per la scheda della feature
* infoTarget: target, se “panel” utilizza MediaWindow su panel ExtJS
* fieldMapping: oggetto per costruire la tabella della scheda (decodifica nomi campi)
  + field: nome del campo
  + label: etichetta da visualizzare

**---**

Modificato metodo OpenLayers.Control.WMSGetFeatureInfo.findLayers per escludere i livelli fuori scala

# Problema popup/select in IE9

**Non si vedono popup.**

**DocStyle: DEVE ESSERE DEFINITO.**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

# Gestione esecuzione funzioni all’avvio

Permette di definire una funzione da eseguire all’avvio della applicazione

Definito namespace **CWN2.OnLoadCallbacksRegistry** che raccoglie le funzioni di callback che si possono eseguire all’avvio della applicazione. Ogni funzione deve prendere in input una eventuale callback.

Definire oggetto onLoadCallback in Configuration.Application che contiene il nome della funzione da eseguire all’avvio (1 sola, se necessarie + di una creare funzioni ad-hoc che fanno la chain delle funzioni da eseguire attraverso un meccanismo di callback) e i parametri richiesti dalla funzione.

Vedi gestione ALFA\_GIS con find per coordinate

# Funzioni di editing

Funzione di editing per geoblog

Vedi:

<http://www.peterrobins.co.uk/it/oledit.html>

<http://ole.geops.de/>

Creato oggetto CWN2.Editor per gestire editing delle feture.

Creato controlli DrawPoint per disegno di punti e DeleteFeature per delete delle feature.

Creato bottoni drawPoint, drawLine,drawPolygon, modifyFeature, dragFeature e deleteFeature per insert geometrie, modifica, spostamento e cancellazione geometrie.

Creata funzione getEditingGeom che ritorna le geometrie del layer di editing in formato GeoJSON

Se passato parametro “WKT” ritorna le geometrie in formato WKT.

# maxZoomLevel

Implementato parametro maxZoomLevel in configurazione mapOptions.

# Lettere accentate in legenda

Bisogna indicare il charset nell’html:

<meta charset="UTF-8">

# Clustering

* <http://openflights.org/blog/2009/10/21/customized-openlayers-cluster-strategies/>
* <http://lists.osgeo.org/pipermail/openlayers-users/2011-February/019481.html>

<http://openlayers.org/dev/examples/strategy-cluster-extended.html>

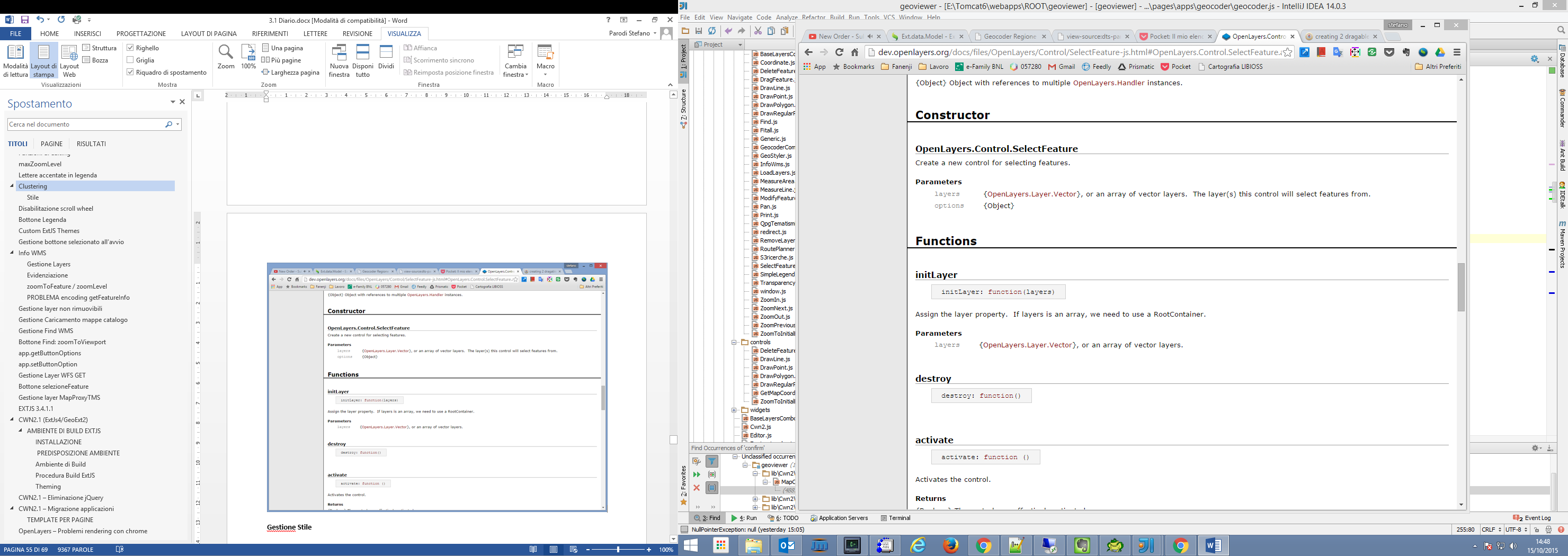
<http://dpinte.wordpress.com/2008/11/07/using-openlayers-filters-with-cluster-strategy/>

**ATTENZIONE! CLUSTERING NON FUNZIONA CON LIVELLI CON FILTRO DEFINITO NELLE CLASSI (LIVELLI MULTICLASSE)**

Quando faccio clustering non vengono soddisfatti i filtri impostati sulle feature. L’operazione di clustering crea delle feature nuove che contengono un attributo (count) e un array di feature (cluster) che contiene le feature singole. Il clustering viene inoltre fatto sul livello generico e in uno stesso cluster potebbero ricadere feature appartenenti a classi diverse.

Se esiste esigenza si creano n layer con classe singola.

Esiste un esempio <http://openlayers.org/dev/examples/strategy-cluster-extended.html> ma non funziona



**Gestione Stile**

In layerFactory. createVectorStyleMap implementata gestione stile cluster.

Feature singola, viene rappresentata come nello stile definito in configurazione.

Feature clusterizzata viene rappresentata con una icona di dimensioni maggiori, il tooltip riporta il numero di feature del cluster.

**Gestione select e popup**

* Feature cluster 🡪 popup con numero elementi e link a funzione di zoom sul cluster
* Feature singola 🡪 popup standard

Parametri:

L’oggetto options nella configurazione della strategia contiene i seguenti parametri che governano il funzionamento del cluster

* Distance: distanza a partire dalla quale si clusterizza, aumentando distance le icone sono meno numerose
* Threshold: numero minimo di feature che possono comparire in un cluster esempio:
  + se 1 possono esistere anche feature songole,
  + se 3 ogni cluster ha almeno 3 feature

## Stile

Cluster eredita stile definito nel layer e modifica la dimensione dell’immagine a seconda del numero di feature clusterizzate e aggiunge un tooltip con il numero.

Oppure utilizza le icone con i numeri (E:\progetti\CartoWebNet2\img\icons\numbers)

# Disabilitazione scroll wheel

Inserito paramtro disableScrollWheel in configurazione application 🡪 mapOptions.

Se impostato a true viene disabilitata la scroll wheel del mouse per lo zoom

# Bottone Legenda

Apertura legenda dentro div mappa sulla destra

# Custom ExtJS Themes

**spket**

<http://www.sencha.com/blog/spket-ide-1611-released-includes-new-ext-theme-builder/>

<http://www.spket.com/>

<http://www.spket.com/ext-theme-builder.html>

Installato plugin su Aptana 3.0 di parodi.datasiel.net

Creato progetto “extjs themes” in aptana

1. Selezionare Perspective Spket II (in alto a destra)
2. From the menu bar, select **File > New > File**
3. In the **File name** field, type **ext.theme** as the name for the new file. Then click **Finish**.
4. Click the **Browse...** button to specify the path to the **Ext Resources** folder which contains css and images:

**E:\progetti\CartoWebNet2\lib\ExtJs\resources**

1. If you check the **Full** option, it will generate "ext-all-**name**.css", otherwise, it will generate "xtheme-**name**.css".
2. In the effect page, adjust the scale to set the color.
3. Click the **Export...** button and specify a folder to save the new resources.

# Gestione bottone selezionato all’avvio

Impostato parametro "pressed" in oggetto “toolbar” deve contenere il nome del bottone da selezionare all’avvio.

Esempio (selezioneterritoriale.js di DownloadDatiRepertorio)

"pressed": "drawRegularPolygon",

# Info WMS

Implementata gestione della info WMS nel bottone “infowms”

## Gestione Layers

Vengono interrogati tutti i **layer di tipo WMS e con attributo queryable = true**

Se è impostata l’opzione “layers” (elenco nome dei layer separati da virgole) nella configurazione del bottone, vengono interrogati solo i layer elencati.

Affinché un layer sia interrogabile deve essere impostato il campo CWR\_LIVELLI.INFO\_URL

## Evidenziazione

La evidenziazione avviene tramite un filtro e una simbologia SLD.

La gestione è nell’oggetto **CWN2.WmsSldHiliter**

**Problema SLD + Symbolscale**

<http://osgeo-org.1560.n6.nabble.com/SYMBOLSCALE-and-SLD-td4270734.html>

Problema Mapserver non ancora risolto.

Eliminato SYMBOLSCALE da generazione Mapfile.

Unica carta che lo utilizza è la 346 (Alberi Monumentali) (livelli 613 / 614)

## zoomToFeature / zoomLevel

Se impostato attributo zoomToSelected=true in opzioni bottone , viene fatto lo zoom sulla feature

Se impostato attributo zoomLevel=XX in opzioni bottone , viene fatto impostato lo zoomLevel a XX

## PROBLEMA encoding getFeatureInfo

**Modificato \_creaMapFileCs.asp impostato Response.Charset = "ISO-8859-1"**

# Gestione layer non rimuovibili

Implementata gestione layer non rimuovibili dalla applicazione.

Basta impostare attributo “unremovable” a true in configurazione del layer.

Vedi corem.js

# Gestione Caricamento mappe catalogo

Gestione della find all’ingresso con evidenziazione e zoom

Creato metodo **CWN2.Application.prototype.loadMap**

Nuovi parametri nelle opzioni di inizializzazione della applicazione:

* **idMap**: codice catalogo della mappa (se impostato viene chiamato il servizio che ritorna la configurazione della mappa)
* **findOptions**: opzioni di find (vedi Gestione Find WMS)

**ATTENZIONE**: eventuali funzioni di callback vengono chiamate in parallelo alla loadMap, la catena di chiamate non viene attualmente gestita dopo la find

La catena di chiamate è:

Costruttore CWN2.Application 🡪 buildApp 🡪 loadMap 🡪 findWMS 🡪CWN2.Util.getWFSBound 🡪 zoomToSelected

Costruttore CWN2.Application 🡪 buildApp 🡪 eventuale callback

**Esempio di utilizzo**

EsempioMappaCatalogo.html:

Input:

ID\_MAP: codice catalogo mappa

# Gestione Find WMS

Gestione della find all’ingresso con evidenziazione e zoom

**PREREQUISITI:**

* **deve essere** **abilitato il servizio WFS**
  + nel caso modificare .asp
* **deve essere utilizzato il campo ID del livello**
  + definito nel mapfile come USING UNIQUE (in genere è il campo ID)
* **nel html deve essere incluso la libreria jQuery xml2json:** 
  + <script src="/cartowebNet2/lib/jQuery/jquery.xml2json.js"></script>

Creato metodo **CWN2.Application.prototype.findWMS**

Prende in input un oggetto di configurazione:

var findOptions = {

layerName: "L3",

idField: "ID",

idList: idList,

zoomLevel: null

};

Dove

* layerName: nome del layer WMS su cui fare la find
* idField: nome del campo UNIQUE ID del layer
* idList: lista dei valori separati da virgola
* zoomLevel: massimo livello di zoom

Utilizza il metodo di utilità CWN2.Util. getWFSBound

Il metodo deve essere richiamato DOPO il caricamento della mappa e del livello su cui fare la find

Creato metodo **CWN2.Util.** **getWFSBound**

Prende in input:

* wfsUrl - {string} URL del servizio WFS
* typeName - {string} nome della feature
* sldFilter - {string} Filtro SLD
* map - {OpenLayers.Map} Mappa OL
* callback - {Function} Funzione di callback da richiamare

Effettua una interrogazione mediante un servizio WFS e ritorna un oggetto OpenLayers.Bounds

**Esempio di utilizzo**

EsempioFindWMS.html:

* carica una mappa in base al codice\_catalogo
* effettua la find

# Bottone Find: zoomToViewport

Gestione della find sulla viewport passata dal servizio google:

Se impostato parametro zoomToViewport nella configurazione del bottone find viene effettuata lo zoom.

# app.getButtonOptions

Funzione generica della Application che ritorna o imposta le opzioni di configurazione di un bottone

Dato il “name” di un bottone cerca nella bottoniera e restituisce le “options”

Esempio: app.getButtonOptions(“selectfeature”);

# app.setButtonOption

Funzione generica della Application imposta una opzione di configurazione di un bottone

Esempio: app.setButtonOption("selectfeature","callBacks",callBacksFunct);

dove callBacksFunct è il nuovo valore della opzione “callBacks”

# Gestione Layer WFS GET

MApserver 5.6 ha un baco per cui i servizi WFS in POST non ritornano le geometrie

Ho implementato la gestione dei layer WFS in modalità GET in layerFactory.

E’ necessario impostare un parametro **method: “GET** nei wfsParams della configurazione del layer.

Es:

"type": "WFS",

"name": "STRISCIATE",

"projection": "EPSG:3003",

"visible": true,

"wfsParams": {

"version": "1.1.0",

"url": url,

"srsName": "EPSG:3003",

"featureType": featureType,

**"method": "GET"**

},

# Bottone selezioneFeature

Bottone generico che permette la selezione da uno o più layer.

**Prerequisiti**:

Configurazione:

* **idLayer** in options della configurazione del bottone (stringa separata da virgole o array)
* **infoIdAttr** e **infoLabelField** in configurazione livelli di selezione (label opzionale)
* **flagSceltaLivello** - se true compare combo con livelli - default false (attributo opzionale)
* **flagSelezioneSingola -** se true permette la scelta di una sola feature – in questo caso ogni selezione annulla la precedente - default false (attributo opzionale)

La configurazione può essere effettuata programmaticamente utilizzando setButtonOption

La selezione funziona sui livelli visibili in legenda.

I risultati della selezione sono visibili su una finestra lista che visualizza campi CODICE/LABEL.

**Funzioni per conferma/annulla**

Implementato meccanismo per definire dinamicamente le funzioni da richiamare alla pressione dei bottoni conferma e annulla.

Le funzioni devono essere definite nella configurazione delle opzioni del bottone (options) nell’oggetto “callBacks”

Accessibile come btnOptions.callBacks[“submit”] e btnOptions.callBacks[“cancel”]

Possono essere impostate in configurazione statica o dinamicamente tramite la funzione setButtonOption

Vedi esempio: “SelezioneHabitat.html”

# Gestione layer MapProxyTMS

Implementata in layerFactory

Vedi <http://parodi.datasiel.net/CartoWebNet2/pages/test/mapproxy.html>

# EXTJS 3.4.1.1

Effettuato upgrade a ExtJS 3.4.1.1 per compatibilità con IE10

# CWN2.1 (ExtJs4/GeoExt2)

<http://docs.sencha.com/extjs/4.2.1>

<http://docs.sencha.com/extjs/4.0.7/#!/guide/upgrade>

<http://www.sencha.com/blog/ext-js-3-to-4-migration/>

<http://parodi.datasiel.net/CartoWebNet2/lib/ExtJs4/examples/compatibility/guides/ext3-to-4-migration.html>

<http://vimeo.com/23027769>

<http://www.sencha.com/forum/showthread.php?124015-Ext-3-to-4-Migration>

<http://stevenschwenke.de/migratingFromExtJS3ToExtJS4>

<http://pratik-poonam.blogspot.it/>

Porting CWN2 a ExtJS4 e GeoExt2

Installato ExtJs 4.2.1 in E:\progetti\CartoWebNet2\lib\ExtJs4

Installato GeoExt2 in E:\progetti\CartoWebNet2\lib\GeoExt2

Creato libreria **CWN2.1**

Creato pagina per porting libreria E:\progetti\CartoWebNet2\pages\test\ExtJS4.html

Creato ambiente di build

Porting applicazioni.

## AMBIENTE DI BUILD EXTJS

### INSTALLAZIONE

Installato SenchaCmd 4.0.1 in E:\Tomcat6\webapps\ROOT\geoviewer\tools

Ruby 1.9.3

Installato SenchaSDK in  [E:\Tomcat6\webapps\ROOT\geoviewer\tools\SenchaSDKTools-2.0.0-beta3](file:///E:\progetti\CartoWebNet2\tools\SenchaSDKTools-2.0.0-beta3) (per creazione immagini, Sencha Cmd non le crea)

Levato SenchaSDK dal path (per non interferire con sencha cmd)

### PREDISPOSIZIONE AMBIENTE

Generato WORKSPACE **ExtJsBuild**per build e theming in E:\Tomcat6\webapps\ROOT\geoviewer\tools\lib\ExtJs\build

cd E:\Tomcat6\webapps\ROOT\geoviewer\tools\lib\ExtJs

sencha -sdk lib/ExtJs4 generate workspace ExtJsBuild

cd ExtJsBuild

Generata APP **builder**

sencha -sdk ext generate app builder builder

<http://dts-parodis.datasiel.net/geoviewer/lib/ExtJs/builder/index.html>

### ****Ambiente di Build****

Copiata dir GeoExt2 in E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder

Aggiunto classpath geoext2 in  E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\.sencha\app\sencha.cfg

app.classpath=${app.dir}/app,${app.dir}/app.js,${app.dir}/GeoExt2/src

Aggiunto skip in E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\.sencha\app\sencha.cfg

skip.sass=true  
skip.resources=true  
skip.slice=true  
skip.theme=true

Aggiunto requires delle classi Ext e GeoExt2 in E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\app.js

### Procedura Build ExtJS

1. Aggiungere requires delle classi Ext necessarie in**E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\app.js**
2. Eseguire procedura **E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\build.bat**

Procedura esegue seguenti operazioni

Genera js di **debug**(testing)

E:

cd E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\

sencha app build -e testing

copy E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\build\testing\builder\app.js E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\ext-debug.js

URL da includere "/cartowebNet2/lib/ExtJs/ext-debug.js"

Genera js di **produzione**(minificato)

cd E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\

sencha app build

copy E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\build\production\builder\app.js E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\ext.js

URL da includere "/cartowebNet2/lib/ExtJsBuild/ext.js"

**Aggiunta Classi**

NOTA: quando si usano nuove classi Ext o GeoExt bisogna

* aggiungere requires delle classi in E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\builder\app.js
* ricompilare il codice con **E:\Tomcat6\webapps\ROOT\geoviewer\lib\ExtJs\build.bat**

### ****Theming****

<http://docs.sencha.com/extjs/4.2.1/#!/guide/theming>

<http://www.rallydev.com/community/engineering/guide-custom-themes-extjs-4>

<https://github.com/NewbridgeGreen/extjs/tree/master/docs/guides/theming>

http://www.sencha.com/blog/understanding-sencha-cmd-packages/

PROCEDURA CREAZIONE CSS

1 - Creazione package

cd E:\progetti\CartoWebNet2\lib\ExtJsBuild\builder

sencha generate theme <TEMA> (crea dir [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto)<TEMA>)

2 - Editing stile

Modificare [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\package.json)<TEMA>[\package.json](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\package.json): indicare css di partenza

Creare [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\sass\var\Component.scss)<TEMA>[\sass\var\Component.scss](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\sass\var\Component.scss): modificare le variabili sass

3 - compilazione css

cd [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto)<TEMA>

sencha package build (genera il file css in  [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)<TEMA>[\build\resources](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)\<TEMA>-all.css)

4 - creazione immagini per IE8

E:\progetti\CartoWebNet2\tools\SenchaSDKTools-2.0.0-beta3\sencha slice theme -d E:\progetti\CartoWebNet2\lib\ExtJs4 -c [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)<TEMA>[\build\resources](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)\<TEMA>-all.css -o [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)<TEMA>[\build\resources](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)\images

genera immagini in [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)<TEMA>[\build\resources](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)\images

5 - copiare i file nella dir degli stili

Creare cartella [E:\progetti\CartoWebNet2\stili\ExtJS](file:///E:\progetti\CartoWebNet2\stili\ExtJS)\resources\<TEMA>\

Copiare tutto da [E:\progetti\CartoWebNet2\lib\ExtJsBuild\packages\](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources)<TEMA>[\build\resources](file:///E:\progetti\CartoWebNet2\ExtJsBuild\packages\rep-carto\build\resources) a [E:\progetti\CartoWebNet2\stili\ExtJS](file:///E:\progetti\CartoWebNet2\stili\ExtJS)\<TEMA>

Includere stile come <link rel="stylesheet" type="text/css" href="/cartowebNet2/stili/extjs/<TEMA>/<TEMA>-all.css" />

NOTA: dopo aver evitato lo stile (punto 2) bisogna completare i punti seguenti

**TODO**: script per create-theme e compile-theme

# CWN2.1 – Eliminazione jQuery

Eliminato jQuery.

CWN2.Util.cloneObj     : jQuery.extend

CWN2.Application.prototype.getButtonOptions/setButtonOptions: $.map (usare Ext.Array.Map)

CWN2.Button.selectfeature: $.each (for)

CWN2.widgetMngr: $.isArray (Ext.isArray)

CWN2.Editor.getGeoJSONGeometry: jQuery.parseJSON (usare Ext.JSON.decode)

CWN2.layerFactory.WMS: $.get (usare Ext.Ajax)

CWN2.Util.getWFSBound: $.xml2json usata da biomassa e esempioFindWms

**ATTENZIONE**: se viene segnalato formato non XML bisogna rigenerare mapfile oppure utilizzare url diretta mapserv.exe

CWN2.Util.getJSON/getJSONP/getXML: $.ajax

Unificata in unica funzione **CWN2.Util.AjaxRequest** con parametri(type,url,urlParams,callBack,args)

# CWN2.1 – Migrazione applicazioni

**Migrate le seguenti app**

* Geoportale
* PTR
* Libioss
* Corem
* Biomassa
* Fototeca

**Modifiche**

* Cambiare stylesheet ExtJs
* Cambiare libreria extjs
* Inserire file per traduzioni ExtJs
* levare libreria GeoExt
* Cambiare libreria CWN2 (CWN2.1)
* Creare Ext.Application e spostare codice da Ext.onReady a metodo launch

## TEMPLATE PER PAGINE

HEAD

<link rel="stylesheet" type="text/css" href="/cartowebNet2/lib/ExtJs4/resources/css/ext-all-gray.css" />  
<link rel="stylesheet" type="text/css" href="/cartowebNet2/stili/default/base.css" />  
  
<script type="text/javascript" src="/cartowebNet2/lib/ExtJsBuild/ext-debug.js"></script>

<script type="text/javascript" src="/cartowebNet2/lib/ExtJsBuild/locale/ext-lang-it.js"></script>  
<script src="http://maps.googleapis.com/maps/api/js?sensor=false&v=3.2&language=it"></script>  
<!--    JQUERY     <script src="/cartowebNet2/lib/jQuery/jquery-1.7.min.js"></script> -->  
<script src="/cartowebNet2/lib/OL/OpenLayers.js" type="text/javascript"></script>  
<script src='/cartowebNet2/lib/proj4js/lib/proj4js-min.js'></script>  
 <script src="/cartowebNet2/lib/Cwn2.1/cwn2-debug.js"></script>  
  
BODY

<script>  
Ext.application({  
     name: 'AppName',

init: function() {

CWN2.loadingScreen = Ext.getBody().mask('Caricamento Applicazione','loadingscreen');

},

launch: function() {  
 // costruzione app CWN2   
 }

});  
</script>

# OpenLayers – Problemi rendering con chrome

Il problema si manifesta con elevato numero di layers e/o dimensione finestra ampia.

Documentato qui:

<https://github.com/openlayers/openlayers/issues/1181>

<https://code.google.com/p/chromium/issues/detail?id=325236>

Workaround

Modificato E:\Tomcat6\webapps\ROOT\geoviewer\lib\OpenLayers\theme\default\style.css

Aggiunta sezione

.olTileImage {

-webkit-backface-visibility: visible !important;

-webkit-transform: none !important;

}

Disabilitazione accelerazione HW gpu