Geocode

Plug-in Guide

2.4.1.130693

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Last Updated

19.09.2012

Table of Contents

1 Introduction

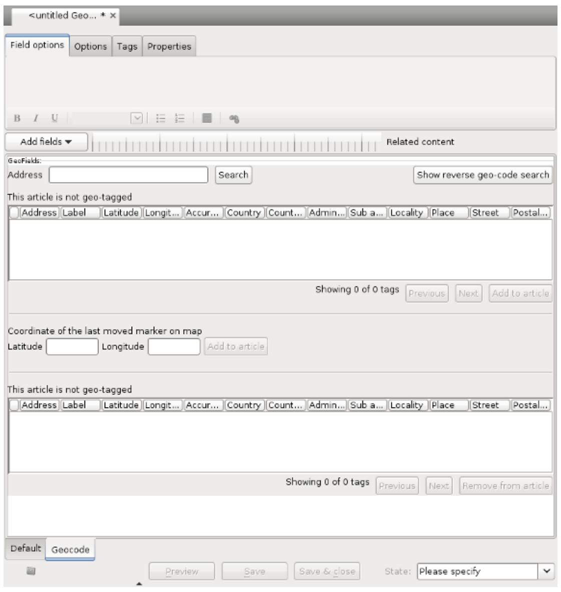
The Geocode plug-in adds easy-to-use geocoding functionality to Escenic Content Engine installations. It allows geographical location records (called **geotags**) to be associated with content items.

With the Geocode plug-in, journalists and editors working in Content Studio are able to easily add geotags to a content item, which are then stored along with the content item in the Content Engine database. Once these tags are stored with content items, they can subsequently be used in a variety of ways:

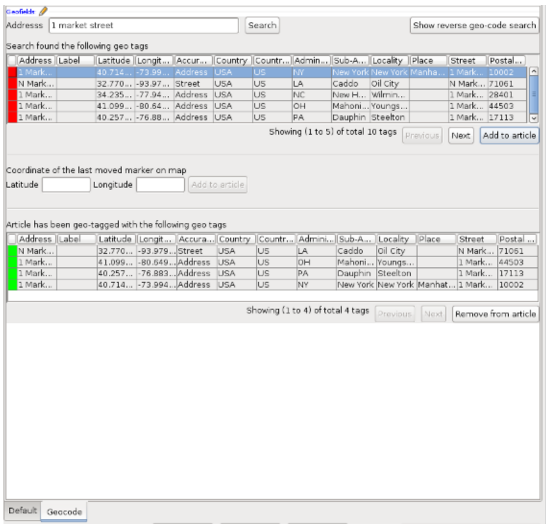
- To search for items in Content Studio by location: either address (city, state, country) or coordinates
- To plot locations on maps
- To create user services that search for news by location

1.1 Using The Geocode Plug-in

When a user creates a "geocodable" content item in Content Studio, the Geocode plug-in is activated, enabling the display of geotags. The geocode field editor of a content item that does not yet have any geotags looks like this:



To add a geotag to a content item you can enter an address in the **Address** field and click on **Search**, in much the same way as if you were looking for an address in an online mapping service. In fact your query is simply passed on to an underlying mapping service such as Google Maps, and the results of the search are displayed in the table immediately below the **Address** field. The results are also displayed in a map browser window as green or red pins.



Address: 1 market street | Search | Show reverse geo-code search

Search found the following geo tags

	Address	Label	Latitude	Longit...	Accur...	Country	Countr...	Admin...	Sub-A...	Locality	Place	Street	Postal...
🟥	1 Mark...		40.714...	-73.99...	Address	USA	US	NY	New York	New York	Manha...	1 Mark...	10002
🟥	N Mark...		32.770...	-93.97...	Street	USA	US	LA	Caddo	Oil City		N Mark...	71061
🟥	1 Mark...		34.235...	-77.94...	Address	USA	US	NC	New H...	Wilmin...		1 Mark...	28401
🟥	1 Mark...		41.099...	-80.64...	Address	USA	US	OH	Mahoni...	Youngs...		1 Mark...	44503
🟥	1 Mark...		40.257...	-76.88...	Address	USA	US	PA	Dauphin	Steelton		1 Mark...	17113

Showing (1 to 5) of total 10 tags | Previous | Next | Add to article

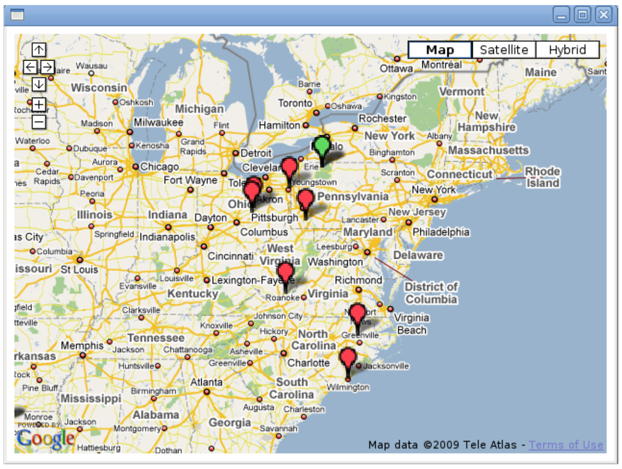
Coordinate of the last moved marker on map

Latitude | Longitude | Add to article

Article has been geo-tagged with the following geo tags

	Address	Label	Latitude	Longit...	Accura...	Country	Countr...	Admini...	Sub-A...	Locality	Place	Street	Postal ...
🟩	N Mark...		32.770...	-93.979...	Street	USA	US	LA	Caddo	Oil City		N Mark...	71061
🟩	1 Mark...		41.099...	-80.649...	Address	USA	US	OH	Mahoni...	Youngs...		1 Mark...	44503
🟩	1 Mark...		40.257...	-76.883...	Address	USA	US	PA	Dauphin	Steelton		1 Mark...	17113
🟩	1 Mark...		40.714...	-73.994...	Address	USA	US	NY	New York	New York	Manhat...	1 Mark...	10002

Showing (1 to 4) of total 4 tags | Previous | Next | Remove from article



A green pin on the map represents a location that matches the search but is not associated with the current content item. A red pin represents a location that matches the search and is associated with the current content item. These locations are also listed in the table at the bottom of the geocode field editor.

To add a geotag to the current content item, simply select one of the locations in the top table and click on the **Add to article** button. To remove a geotag, select one of the locations in the bottom table and click o the **Remove from article** button. Note that selecting a location in either of the tables will cause the map browser to zoom in on the corresponding pin.

If you have several geocoded content items open in Content Studio, then then the geocode field editor displays the tags associated with the currently selected content item. Once you select the geocode field editor, the map browser is also updated to display the same set of tags.

You can also search for geotags by coordinate using the **Show reverse geo-code search** function. You enter a latitude, a longitude and a radius, and all the geotags in the circle defined by these numbers are displayed in the results table.

2 Installation

The following preconditions must be met before you can install Geocode 2.4.1.130693:

- The Content Engine is installed and in working order.
- The Escenic assembly tool has been extracted and successfully used to set up a test EAR file as described in the **Escenic Content Engine Installation Guide**.
- The Escenic Lucy plug-in is installed and in working order.
- You have the correct distribution file (**`geocode-dist-2.4.1.130693.zip`**).
- If you intend to use the default service provider implementation included with the Geocode plug-in then you need a Google Map API key to be able to access Google Maps. To get a key you need to register as a user of the API with Google.

Plug-ins must be installed using the Escenic assembly tool.

2.1 Install Geocode

In the following description, *escenic-home* refers to the server folder in which the Content Engine is installed.

Installing Lucy on the server involves the following steps:

1. **Make sure there is a plug-in folder:** If the folder *escenic-home*/**`plugins`** does not already exist on your server, create it. If for some reason you need to create the plug-in folder in some other location, edit the *escenic-home*/**`assemblytool/assemble.properties`** file and set the **`plugins`** property accordingly. For example:

   ```
   plugins=escenic-home/my/plugin/folder
   ```

   This folder will be referred to as *plugin-home* in the rest of this manual.
2. **Unpack the Geocode distribution:** Unpack the Geocode distribution file to *plugin-home*. This will result in the creation of a *plugin-home*/**`geocode`** folder.
3. **Configure Geocode:** See section 2.1.1 for details.
4. **Rebuild and deploy the Content Engine:** Build the Escenic enterprise archive by entering the following commands:

   ```
   cd escenic-home/assemblytool
   ant ear
   ```

   The assembly tool will then add the Geocode plug-in to the Content Engine's classpath, including default configuration files and any required web application components. Deploy the new EAR file. (For general instructions on how to deploy the EAR file on different application servers, see the **Escenic Content Engine Installation Guide**.)

5. **Verify the plug-in installation:** See section 2.2 for details of how to verify plug-in installations.

If the application server does not support EAR-based deployment, then all the JAR files located in the *plugin-home*`/geocode/lib` folder must be added to the application server's classpath. All the WAR files that have been rebuilt with the assembly tool should be redeployed.

2.1.1 Configuration

The following sections describe how to configure the Geocode plug-in. In these instructions, the placeholder *escenic-config* is used to represent the path of your Escenic configuration, as defined with the `com.escenic.config` property in *escenic-home*`/assemblytool/assemble.properties`. If `com.escenic.config` is not defined, then *escenic-config* has a default definition of *escenic-home*`/localconfig`.

2.1.1.1 Configure the Solr search engine

Geocode requires some geocode-specific fields to be indexed by the Solr search engine. In order to ensure that this is done, you must:

1. Open the Solr `schema.xml` file for editing. This file is usually located in the `${solr.solr.home}/conf` folder.
2. Add the following element as a child of the `fields` element:

```
<dynamicField name="geocode.*" type="string" indexed="true" stored="true" multiValued="true"/>
```

3. Save the file and exit.

2.1.1.2 Set the Lucy SearchEngine Properties

By default, Lucy (which is used by the Geocode plug-in) looks for its search engine at `http://localhost:8080/solr`, which is the location of the default Lucy search engine, called `LucySearchEngine`. If this is not the Solr engine you want Lucy to use, then you will need to:

1. Copy `SearchEngine.properties` from *escenic-home*`/engine/plugins/geocode/misc/siteconfig/com/escenic/geocode/search/lucy/SearchEngine.properties` to *escenic-config*`/com/escenic/geocode/search/lucy/SearchEngine.properties`.
2. Open the copied file for editing.
3. Set the `solrURI` property to point to the correct Solr engine location. For example:

```
solrURI=http://mysolrhost:8080/solr
```

4. Save the file and exit.

2.1.1.3 Configure the plug-in to use a Proxy Server

Content Studio uses whatever proxy is defined for the browser used to start it, and by default Geocode uses the same proxy (if any). If, however, you want the Geocode plug-in to use a particular proxy server when accessing external domains, you can configure it to do so. The Geocode browser component displayed in Content Studio will then also direct its requests via the same proxy server. To configure a proxy server for the Geocode plug-in:

1. Copy **`Proxy.properties`** from *escenic-home*/**`engine/plugins/geocode/misc/siteconfig/com/escenic/geocode/conf/Proxy.properties`** to *escenic-config*/**`com/escenic/geocode/conf/Proxy.properties`**.
2. Open the copied file for editing.
3. Set the **`host`** property to point to the required host name or IP address. For example:

   ```
   host=myproxy.mydomain.com
   ```
4. Set the **`port`** property to point to the required port on the proxy. For example:

   ```
   port=7777
   ```
5. Save the file and exit.

2.1.1.4 Add geocode fields to content types

In order to make content items "geocodable" you must add a special geocode field to their content type definitions in the **`content-type`** resource. The following example shows what such a field definition should look like:

```
<content-types xmlns="http://xmlns.escenic.com/2008/content-type"
               xmlns:geocode="http://xmlns.escenic.com/2009/studio/plugin/geocode"
               xmlns:ui="http://xmlns.escenic.com/2008/interface-hints"
               version="4">
  <!-- Please note that geocode namespace declared above -->
  ...
  <content-type name="restaurant-review">
    ...
    <panel name="geocode">
      ...
      <field type="basic" name="com.escenic.geocode" mime-type="text/plain">
        <geocode:geocode-editor enabled="true">
          <geocode:service-provider>
            com.escenic.geocode.service.impls.GoogleServiceImpl
          </geocode:service-provider>
          <geocode:map-provider>
            com.escenic.geocode.studio.mapviewer.impls.GoogleMapsViewerImpl
          </geocode:map-provider>
          <geocode:google-api>
            <geocode:api-key><!-- your Google API key --></geocode:api-key>
            <geocode:registered-url><!-- your Google API registration URL --></geocode:registered-
url>
          </geocode:google-api>
        </geocode:geocode-editor>
      </field>
    </panel>
    ...
  </content-type>
  ...
</content-types>
```

You must add one such field definition to each content type that you want to be able to geocode.

Note the following:

- A geocode field must have the name **`com.escenic.geocode`** (with all characters in lower case).

- Geocode fields contain elements belonging to a special **`geocode`** XML namespace. You must therefore add a declaration for this namespace to your **`content-type`** resource (see the bold text in the example above).

The **`geocode`** namespace elements to be included in the field definition are:

`geocode:geocode-editor`
: This element specifies that the field is a geocode editor field, an contains all the other elements. The **`enabled`** attribute must be set to **`true`**.

`geocode:service-provider`
: Must be set to the fully qualified name of a class that implements the abstract class **`com.escenic.geocode.service.api.GeocodeService`** and **`com.escenic.geocode.service.api.ReverseGeocodeService`**. The specified class is instantiated and used for address- and coordinate-based searches performed by the user. The default implementation shipped with the plug-in is **`com.escenic.geocode.service.impls.GoogleServiceImpl`**, which makes use of services provided by Google.

`geocode:map-provider`
: Must be set to the fully qualified name of a class that implements the abstract class **`com.escenic.geocode.studio.mapviewer.api.AbstractMapViewer`** The specified class is instantiated and used to display maps in Content Studio. The map viewer implementation displays geotag markers on the map, and supports drag and drop etc. The default implementation shipped with the plug-in is **`com.escenic.geocode.studio.mapviewer.impls.GoogleMapsViewerImpl`**, which displays Google maps.

`geocode:api-key`
: If you are using the default **`geocode:service-provider`**, then the value you enter here must be a Google API key, obtained by registering with Google as a user of their mapping API.

`geocode:registered-url`
: If you are using the default **`geocode:service-provider`**, then the value you enter here must be the URL for which the specified Google API key is valid.

If you have implemented your own service provider class that makes use of a different mapping service, then you can replace the **`api-key`** and **`registered-url`** elements with other elements holding whatever identification parameters are required by your chosen service, and program your service provider class to retrieve and use the parameters as required.

For general information about the **`content-type`** resource and how to edit it, see the **Escenic Content Engine Resource Reference**.

2.2 Verify The Installation

To verify the status of the Lucy installation, open the Escenic Admin web application (usually located at **`http://`***server***`/escenic-admin`**) and click on **View installed plugins**. The status of the plug-ins is indicated as follows.

✓
The plug-in is correctly installed.

✗
The plug-in is not correctly installed.

3 Using the Geocode Field in Templates

To help the template developers retrieve geocoded articles, the plug-in ships with some struts Action and ActionForm.

com.escenic.geocode.publication.struts.GeoArticleSearchAction: the Action class that can be used to search for geocoded Articles

com.escenic.geocode.publication.struts.GeoArticleSearchForm: the ActionForm used by the action class mentioned above.

The form and the action class can be configured in your struts-config.xml as follows:

```
<struts-config>
  <form-beans>
    <form-bean name="com.escenic.geocode.presentation.struts.GeoArticleSearchForm"
               type="com.escenic.geocode.presentation.struts.GeoArticleSearchForm"/>
  </form-beans>
  <action-mappings>
    <action path="/geoArticleSearch"
            type="com.escenic.geocode.presentation.struts.GeoArticleSearchAction"
            name="com.escenic.geocode.presentation.struts.GeoArticleSearchForm"
            scope="request"
            validate="false"
            parameter="actionCommand">
      <forward name="success" path="/template/common.jsp"/>
      <forward name="error" path="/template/common.jsp"/>
    </action>
  </action-mappings>
  <message-resources parameter="com.escenic.geocode.Resources"/>
</struts-config>
```

3.1 Form Properties

A brief explanation of the form properties is as follows:

publicationIds: the publications in which the action should search for articles

searchExpression: the search text which is looked in the article fields

fromLattitude: the latitude from which articles should be looked for

fromLongitude: the longitude from which articles should be looked for

toLattitude: the latitude to which articles should be looked for

toLongitude: the longitude to which articles should be looked for

startDay: the start day from which articles should be searched for

startMonth: the start month of the from date

startYear: the start year of the from date

toDay: the day of the to date

toMonth: the month of the to date

toYear: the year of the to date

sectionIds: the ids of the section in which the articles should be searched for

searchSubSectionAllowed: indicates if articles in subsections of sections specified by **sectionIds** should also be searched

articleTypes: restricts the searched article types

Please see javadoc for further details of the form properties.

An example war file is shipped with the plug-in to display the minimum functionality of the plug-in in publication templates.

4 Writing your own map service

Geocode plug-in allows you to write your own map services and plug them into the plug-in. The plug-in provides some java interfaces. Any class implementing them can be used with the plug-in to provide the functionalities. Below is a set of interfaces which can be implemented to provide custom map services.

com.escenic.geocode.service.api.GeocodeService

The interface declares the methods required by the plug-in to allow a user to search for geo tags based on adress. Once a concrete implementation of the interface is ready, it can be configured in the article type definition.

com.escenic.geocode.service.api.ReverseGeocodeService

The interface declares the methods required by the plug-in to allow a user to do a reverse geo search. This allows a user to search for nearby geo tagged places with a given location which consists of latitude and longitude.

com.escenic.geocode.studio.mapviewer.api.AbstractMapViewer

The abstract class defines the methods necessary to display a custom map in Geocode plug-in.

Example configuration

```
<field type="basic" name="com.escenic.geocode" mime-type="text/plain">
<geocode:geocode-editor enabled="true">
  <geocode:service-provider>test.MyGeocodeService</geocode:service-provider>
  <geocode:map-provider>test.MyCustomMap</geocode:map-provider>
  <geocode:my-api>
    <geocode:my-key>tester007</geocode:my-key>
    <geocode:my-password>band</geocode:my-password>
  </geocode:my-api>
</geocode:geocode-editor>
</field>
```

Please see the javadoc to get an overview on the interfaces and classes.