# GeoIP User Guide

## Start working and database update

When extension is successfully installed you can start work by updating pre-installed databases. Go to Store > Configuration > ToBai > GeoIP2. You will see there "GeoIP2Lite databases update" and "GeoIP2 WebService" tabs. When you open "GeoIP2Lite databases update" tab you will see there multiselect dropdown "Choose Database For Update". There you can choose one or both available databases, Save Config (orange button in the right top corner) and only then press Update button to update it immediately . If you don't Save Config the update will not apply to correct databases!  In a few seconds the chosen databases will be updated and the date of last updated will change. We should emphasize that if you update GeoIP2 database too often you may be banned for several hours.

### Cron

You also can set to update databases via cron. Just set "Update" by Cron" to "Yes" and configure update intervals in the file crontab.xml in your extension. You should add this code:

```
<group id="default">
    <job name="tobai_geoip2_update_db"
instance="Tobai\GeoIp2\Model\Observer"
method="updateDb">
        <schedule>0 0 * * 5</schedule>
    </job>
</group>
```

## Using Data from Databases

Start working with the extension us very easy. Lets discover simple example where you need to check country code in your Controller.

You should add object \Tobai\GeoIp2\Model\CountryInterface $country to your Controller and it is ready to be used:

```
<?php
namespace <VendorName>\<ModuleName>\Controller\Index;

use Magento\Framework\App\Action\Context;
use Tobai\GeoIp2;

class  ControllerName extends
\Magento\Framework\App\Action\Action
{
    /**
 * @var \Tobai\GeoIp2\Model\CountryInterface
 */
  protected $country;

  /**
 * @param Context $context
 * @param \Tobai\GeoIp2\Model\CountryInterface $country
 */
  public function __construct(
        Context $context,
  GeoIp2\Model\CountryInterface $country
  )
     {
        parent::__construct($context);
  $this->country = $country;
  }

    public function execute()
    {
        $countryCode = $this->country->getCountryCode();
   var_dump($countryCode); // result: string 'UA'
(length=2)
    }
}
```

You can also use country object to extract for example such data:

```
$country = $this->country->getCountry();
var_dump($country->country->isoCode); // string 'UA'
(length=2)
var_dump($country->country->geonameId); // int 690791
var_dump($country->country->names['es']); // string
'Ucrania' (length=7)
```

$country->country->names is an array of local country names and can be used appropriately.

To use City database you should a little bit more manipulations. Together with Controller you should add one line your di.xml file:

```
<preference for="Tobai\GeoIp2\Model\CountryInterface"
type="Tobai\GeoIp2\Model\Database\Data\City" />
```

City database is bigger and contains a lot more data. Here some examples:

```
$city = $this->city->getCity();
var_dump($city->city->names['de']); //string 'New York
City' (length=13)
var_dump($city->location->latitude); //float 40.7317
var_dump($city->location->longitude); //float -73.9885
var_dump($city->location->metroCode); //int 501
var_dump($city->location->timeZone); //string
'America/New_York' (length=16)
```

There are also other fields available:

```
$city->postal
$city->subdivisions
$city->continent
$city->country
```

To see the full object just make var_dump($city);

## Adding Custom Database

It is very easy to add a new custom database. First of all you should create a new Magento extension or use some existing one because it is not a good idea to edit our module due to future extension updates.

Your extension should put ToBai_GeoIP2 in sequence. You should edit file <your Magento install dir>/app/code/<VendorName>/<ModuleName>/module.xml:

```
<config
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../../../../../lib/intern
al/Magento/Framework/Module/etc/module.xsd">
    <module name="Tobai_StoreGeoIp"
setup_version="0.1.0">
        <sequence>
            <module name="Tobai_GeoIp2"/> <!-- add this
line -->
            <module name="Magento_Store"/>
        </sequence>
    </module>
</config>
```

So in this extension you should open di.xml file which should be located here <your Magento install dir>/app/code/<VendorName>/<ModuleName>. There you should add this code:

```
<type name="Tobai\GeoIp2\Model\Database">
    <arguments>
        <argument name="databases" xsi:type="array">
            <item name="country"
xsi:type="string">GeoLite2-Country.mmdb</item>
            <item name="city"
xsi:type="string">GeoLite2-City.mmdb</item>
            <item name="your_database"
xsi:type="string">Your_Database.mmdb</item> <!-- this is
the name of a file with your custom database -->
        </argument>
    </arguments>
</type>
```

Then you should copy the file with your database to directory var/tobai-geoip/Your_Database.mmdb.

If you plan to update this database using ToBai GeoIP2 updater you should also add this code to your di.xml file which should be located here <your Magento install dir>/app/code/<VendorName>/<ModuleName>:

```
<type
name="Tobai\GeoIp2\Model\System\Config\Source\AvailableD
b">
    <arguments>
        <argument name="databases" xsi:type="array">
            <item name="country"
xsi:type="string"><![CDATA[Country database]]></item>
            <item name="city"
xsi:type="string"><![CDATA[City database]]></item>
            <item name="your_database"
xsi:type="string"><![CDATA[Your Database]]></item>
        </argument>
    </arguments>
</type>
<type name="Tobai\GeoIp2\Model\Database\Updater">
    <arguments>
        <argument name="archive"
xsi:type="object">Magento\Framework\Archive\Gz</argument
>
        <argument name="dbLocation"
xsi:type="string">http://geolite.maxmind.com/download/ge
oip/database/%db_name%.gz</argument> <!-- link to the
external database for update -->
        <argument name="dbArchiveExt"
xsi:type="string">.gz</argument>
    </arguments>
</type>
```

You should provide your link to database which will be used for update. But when you change the default link then pre-installed databases Country and City won't be able to update.

## Using Custom Database

You have to edit your di.xml file and add following code to it:

```
<type name="Tobai\GeoIp2\Model\Database">
    <arguments>
        <argument name="databases" xsi:type="array">
            <item name="country"
xsi:type="string">GeoLite2-Country.mmdb</item> <!-- it
is pre-installed database, you should also declare it
here if you want still use it -->
            <item name="city"
xsi:type="string">GeoLite2-City.mmdb</item> <!-- it is
pre-installed database, you should also declare it here
if you want still use it -->
            <item name="your_database"
xsi:type="string">Test-City.mmdb</item> <!-- this is the
name of a file with your custom database -->
        </argument>
    </arguments>
</type>
```

Declare $readerFactory at constructor:

```
/**
 * @var \GeoIp2\Database\Reader
 */
protected $reader;


/**
 * @param \Tobai\GeoIp2\Model\CountryInterface $city
 */
public function __construct(
           Tobai\GeoIp2\Model\Database\ReaderFactory
$readerFactory
) {
  $this->readerFactory = $readerFactory;
}
```

Now you can use your custom database:

```
$yourDatabase =
$this->readerFactory->create('your_database');
var_dump($yourDatabase->city->names);
```

## GeoIP2 WebService

You can set Web Service configuration in extension admin just click at "GeoIP2 WebService".
There are fields:

**User Id** - login to the API server

**License Key**

**Host** - the one you connect to communicate with API

Just fill in the fields and Save Config.

### Usage

First you should declare $clientFactory at constructor:

```
/**
 * @var \Tobai\GeoIp2\Model\WebService\ClientFactory
 */
protected $clientFactory;

/**
 * @param \Tobai\GeoIp2\Model\CountryInterface $city
 */
public function __construct(
    ClientFactory $clientFactory
) {
    $this->clientFactory = $clientFactory;
}
```

Then you should create a Client:

```
$client = $this->clientFactory->create();
```

This way you will request remote API using settings declared in extension configuration:

You can also declare new request settings directly in the client creation, just add config array:

```
$configs = array(
    'userId' => 'yourId',
    'licenseKey' =>
'0:2:R8xXYci5dwNUbPnSXlSY9LEFQlyVELUc:5At1XcbBXuNbuLe01L
PcCKdFC7jqKxq4ioQgesocAHU=',
    'host' => 'geoip.maxmind.com'
);
$client = $this->clientFactory->create($configs);
```

Here is an usage example of 'City' WebService:

```
$client = $this->clientFactory->create();
$record = $client->city('128.101.101.101');

print($record->country->isoCode . "\n"); // 'US'
print($record->country->name . "\n"); // 'United States'
print($record->country->names['zh-CN'] . "\n"); // '??'
print($record->mostSpecificSubdivision->name . "\n"); //
'Minnesota'
print($record->mostSpecificSubdivision->isoCode . "\n");
// 'MN'
print($record->city->name . "\n"); // 'Minneapolis'
print($record->postal->code . "\n"); // '55455'
print($record->location->latitude . "\n"); // 44.9733
print($record->location->longitude . "\n"); // -93.2323
```