Configuring data sources in Ushahidi V3

Ushahidi is all about data.. a beautiful map is only useful if you can get the data or messages you want and make them in to posts.

In Ushahidi V3 SMS, Twitter, Email, RSS and other incoming data is all handled through data providers. Each data provider plugin handles a different source (ie. SMSSync, Twillio, etc). We currently have support for SMS and Email message types, and plugins for email, twilio, smssync and nexmo data providers.

Enabling and configuring data providers is handled through the data-provider.php config file. By default all data providers are disabled, you'll need to enable and configure the ones you need. The base data-provider.php looks like this:

application/config/data-provider.php

```
<?php defined('SYSPATH') OR die('No direct script access.');

/**
 * Ushahidi Settings
 *
 * @author     Ushahidi Team <team@ushahidi.com>
 * @package    Ushahidi\Application\Config
 * @copyright  2013 Ushahidi
 * @license    https://www.gnu.org/licenses/agpl-3.0.html GNU Affero General Public License Version 3 (AGPL3)
 */

return array(
    'default_providers' => array(
        Message_Type::SMS => 'smssync',
        Message_Type::IVR => FALSE,
        Message_Type::EMAIL => 'email',
        Message_Type::TWITTER => 'twitter'
    ),
    'providers' => array(
        'smssync' => FALSE,
        'email' => FALSE,
        'twilio' => FALSE,
        'nexmo' => FALSE,
    ),
    /*'nexmo' => array(
        'from' => '',
        'api_key' => '',
        'api_secret' => ''
    ),*/
    /*'twilio' => array(
        'from' => '',
        'account_sid' => '',
        'auth_token' => '',
        'sms_auto_response' => ''
    ),*/
    /*'smssync' => array(
        'from' => '12345',
        'secret' => '1234'
    ),*/
    /*'email' => array(
        'from' => 'test@robbiemackay.com',
        'from_name' => 'tester',

        'incoming_type' => '',
        'incoming_server' => '',
```

```
    'incoming_port' => '',
    'incoming_security' => '',
    'incoming_username' => '',
    'incoming_password' => '',

    'outgoing_type' => '',
    'outgoing_server' => '',
    'outgoing_port' => '',
    'outgoing_security' => '',
    'outgoing_username' => '',
    'outgoing_password' => ''
)*/
```

```
);
```

1. To start configuring data providers, first copy this file from application/config/data-provider.php to application/config/environments/development/data-provider.php
2. Edit application/config/environments/development/data-provider.php and enable the providers you need. Here's an example config to enable SMSSync and email providers.

application/config/environments/development/data-provider.php

```
<?php defined('SYSPATH') OR die('No direct script access.');

return array(
    'default_providers' => array(
        Message_Type::SMS => 'smssync', // If you're using a different provider you should change this default provider for sending SMS.
        Message_Type::EMAIL => 'email',
    ),
    'providers' => array(
        'smssync' => TRUE,
        'email' => TRUE,
    ),
    'smssync' => array(
        'from' => '12345', // This should be the mobile number we send from
        'secret' => '1234' // Put this secret key in the SMSSync settings on your phone
    ),
    'email' => array(
        'from' => 'test@gmail.com',
        'from_name' => 'tester',

        'incoming_type' => 'pop3',
        'incoming_server' => 'pop.gmail.com',
        'incoming_port' => '995',
        'incoming_security' => 'ssl',
        'incoming_username' => 'test@gmail.com',
        'incoming_password' => 'sometestpassword',

        'outgoing_type' => 'smtp',
        'outgoing_server' => 'smtp.gmail.com',
        'outgoing_port' => '465',
        'outgoing_security' => 'ssl',
        'outgoing_username' => 'test@gmail.com',
        'outgoing_password' => 'sometestpassword'
    )
);
```

3. Once you've configured providers, run
   ./minion DataProvider:incoming - to check incoming messages
   ./minion DataProvider:outgoing - to send outgoing messages

For a real deployment you probably want to set up a cronjob to run ./minion DataProvider:incoming regularly.

At the moment there isn't any UI for sending outgoing messages, so there's not much use creating a cronjob for that.