Versioning

If you build your API unversioned, it’s terrifying. Let’s imagine you’re pushing out a breaking change -  
basically any change that runs counter to what client developers have planned for, such as renaming or  
deleting a parameter or changing the format of the response - you run the risk of bringing down many, if  
not all, of your customers’ systems, leading to angry support calls or, worse, massive churn. That’s why  
you have to keep your API versioned. In Yii2, versioning can be easily done through modules, so versions  
will be represented as isolated block of code.

Getting ready

Repeat all steps from the Creating a REST server recipe’s Getting ready and How to do it... sections.

How to do it...

1. Create the following structure in your app folder. In total, you have to create the @app/modules  
   folder with the vi and v2 folders inside it. In each module’s folder, you must create controllers and  
   models folders:

app/

modules/

vi/

controllers/

FilmController.php  
Module.php

v2/

controllers/

FilmController.php  
Module.php

1. Add the import modules to @app/config/web. php:

'modules' => [

'v1' => [

'class' => 'app\modules\v1\Module',

],

' v2' => [

'class' => 'app\modules\v2\Module'

]

],

1. Create @app/modules/v1/controllers/FilmController.php and  
   @app/modules/v2/controllers/FilmController . php with the following code:

<?php

namespace app\modules\v1\controllers;  
use yii\rest\ActiveController;

class FilmController extends ActiveController  
{

public $modelClass = 'app\models\Film';

}

<?php

namespace app\modules\v1\controllers;  
use yii\rest\ActiveController;

class FilmController extends ActiveController  
{

public $modelClass = 'app\models\Film';

}

<?php

namespace app\modules\v1;

class Module extends \yii\base\Module  
{

public function init()

{

parent::init();

}

}

<?php

namespace app\modules\v2;

class Module extends \yii\base\Module  
{

public function init()

{

parent::init();

}

}

Create @app/modules/v1/Module . php and @app/modules/v2/Module . php with the following code:

How it works...

Each module represents an independent version of our API.

Now you will be able to specify the API’s version in two ways:

1. By the API’s URL. You can specify either v1 or v2 versions. The result is that [http://yii-  
   book.app/v1/film](http://yii-book.app/v1/film) will return a list of films for version 1 and <http://yii-book.app/v2/film> will  
   do so for version 2.
2. You can also put a version number through HTTP request headers. As usual, it can be done through  
   the Accept header:

// as a vendor content type

Accept: application/vnd.company.myproject-v1+j son  
// via a parameter

Accept: application/json; version=v1

So, we now have two versions of our API, and we can easily modify the v2 version without any  
headaches. Our old customers continue to work with the v1 version, and new customers or those who

would like to upgrade will use the v2 version.

There’s more...

Fur further information, refer to:

* <http://www.yiiframework.com/doc-2.0/guide-rest-versioning.html>
* <http://budiirawan.com/setup-restful-api-yii2/>