Adding fwconsole commands

Introduction

This page covers making command classes for fwconsole. The fwconsole is built with symfony console and their documentation can be referenced for using symfony internals.

Skeleton File

The class should be $WEBROOT/admin/modules/$MODULE/Console/Classname.class.php

```
<?php
//Namespace should be FreePBX\Console\Command
namespace FreePBX\Console\Command;
//Symfony stuff all needed add these
use Symfony\Component\Console\Command\Command;
use Symfony\Component\Console\Input\InputArgument;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Input\InputOption;
use Symfony\Component\Console\Output\OutputInterface;
//Start class. Class name should be same as file name. Helloworld.class.php
class Helloworld extends Command {
        //Declare component and your options.
        protected function configure(){
                $this->setName('helloworld')
                ->setAliases(array('hw'))
                ->setDescription('This says hello to the world')
                ->setDefinition(array(
                        new InputOption('flag', 'f', InputOption::
VALUE_NONE, 'We are setting F flag'),
                        new InputOption('dlag', 'd', InputOption::
VALUE_REQUIRED | InputOption::VALUE_IS_ARRAY, 'We are setting D flag'),
                        new InputArgument('args', InputArgument::IS_ARRAY,
'farrrrrrgs', null),))
                ->setHelp('Long Help Message here <info>FORMATTED STRING<
/info>');
        }
        protected function execute(InputInterface $input, OutputInterface
$output){
                $arg = $input->getArgument('args');
                $dlag = $input->getOption('dlag');
                if ($input->getOption('flag')) {
                        $text = "Flag Set.";
                } else {
                        $text = "No Flag Set";
                }
                $output->writeln($text);
                if($dlag){ print_r($dlag);}
        }

}
```

Namespace

All Commands should be in the namespace: FreePBX\Console\Command

```
namespace FreePBX\Console\Command;
```

Includes

These are the base includes. You may have others depending on the components you use.

```
use Symfony\Component\Console\Command\Command;
use Symfony\Component\Console\Input\InputArgument;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Input\InputOption;
use Symfony\Component\Console\Output\OutputInterface;
```

Requires Methods

configure

You **must** have at a minimum InputArgument because your command is arg 0

fwconfig mycommand bar

array:

0 => mycommand

1 => bar

```
protected function configure(){
                $this->setName('helloworld')
                ->setAliases(array('hw'))
                ->setDescription('This says hello to the world')
                ->setDefinition(array(
                        new InputOption('flag', 'f', InputOption::
VALUE_NONE, 'We are setting F flag'),
                        new InputOption('dlag', 'd', InputOption::
VALUE_REQUIRED | InputOption::VALUE_IS_ARRAY, 'We are setting D flag'),
                        new InputArgument('args', InputArgument::IS_ARRAY,
'farrrrrrgs', null),))
                ->setHelp('Long Help Message here <info>FORMATTED STRING<
/info>');
        }
```

Flag(option) syntax:

InputOption([longname],[shortcut],[Type],[help text])

You can set 3 types of flags:

Flag no argument
`new InputOption('longname', 'l', InputOption::VALUE_NONE, 'Help Text');`

Single use flag with arg

```
new InputOption('longname', 'l', InputOption::VALUE_REQUIRED, 'Help Text')
```

Multi-use flag with argument

```
new InputOption('longname', 'l', InputOption::VALUE_REQUIRED |
InputOption::VALUE_IS_ARRAY, 'Help Text')
```

execute

```
        protected function execute(InputInterface $input, OutputInterface
$output){
                $arg = $input->getArgument('args');
                $dlag = $input->getOption('flag');
                if ($input->getOption('flag')) {
                        $text = "Flag Set.";
                } else {
                        $text = "No Flag Set";
                }
                $output->writeln($text);
                if($dlag){ print_r($dlag);}
        }
```

Module XML Lazy Loading

Starting in Framework 14.0.8 and Framework 15.0.7 Console commands now support "Lazy Loading". The Symfony Console (which is what FreePBX uses to do console commands) suffered a shortcoming since day one: you must instantiate all commands to register them in the console application. The reason is that even the command name itself is defined inside a method called configure(), so you must instantiate the command class to configure it and get the command name.

This means that for every "command" that was directly run the system would still load every single OTHER console command. With module XML loading this is not longer the case (and don't worry the system will fall back to the older method easily!)

See more here: https://symfony.com/blog/new-in-symfony-3-4-lazy-commands

To get this working just add a console block to the module.xml file as follows:

```
<console>
        <command>
                <name>endpoint<name>
                <alias>epm</alias>
                <alias>ep</alias>
                <class>Endpoint</class>
        </command>
</console>
```

- **name:** The name of the command, MUST be the same name you use in the method setName()

- **alias(es):** The name of the aliases for this command, MUST be the same name(s) you set in the array of setAlias()
- **class:** The name of the class file in the console/ folder to load for this. If not set then name is assumed. Resulting format would be: rawname/Console/<class>.class.php
    - EX: rawname/Console/Endpoint.class.php

Hooks

Hooking in to fwconfig start/stop

module.xml

```
        <hooks>
                <framework namespace="FreePBX\Console\Command" class="
Start">
                        <method class="Dahdiconfig" callingMethod="
preAsteriskHooks">preAsteriskHooks</method>
                        <method class="Dahdiconfig" callingMethod="
postAsteriskHooks">postAsteriskHooks</method>
                </framework>
                <framework namespace="FreePBX\Console\Command" class="Stop"
>
                        <method class="Classname" callingMethod="
preAsteriskHooks">preAsteriskHooks</method>
                        <method class="Classname" callingMethod="
postAsteriskHooks">postAsteriskHooks</method>
                </framework>
        </hooks>
```

Classname.class.php

```
  public function preAsteriskHooks($output) {
    $output->writeln("This is from Classname");
  }
  public function postAsteriskHooks($output) {
    $output->writeln("This is from Classname");
  }
```