IOC container

beyond constructor injection

#laravelcologne @hannesvdvreken

Hi, my name is Hannes.

madewithlove.be

Administration Area

Personal

https://madewithlove.be/admin/backend/auth/signin

OCTOBER

Getting back to basics

login

password

Login

Forgot your password?

IoC Container

CONSTRUCTOR INJECTION

illuminate/container

league/container

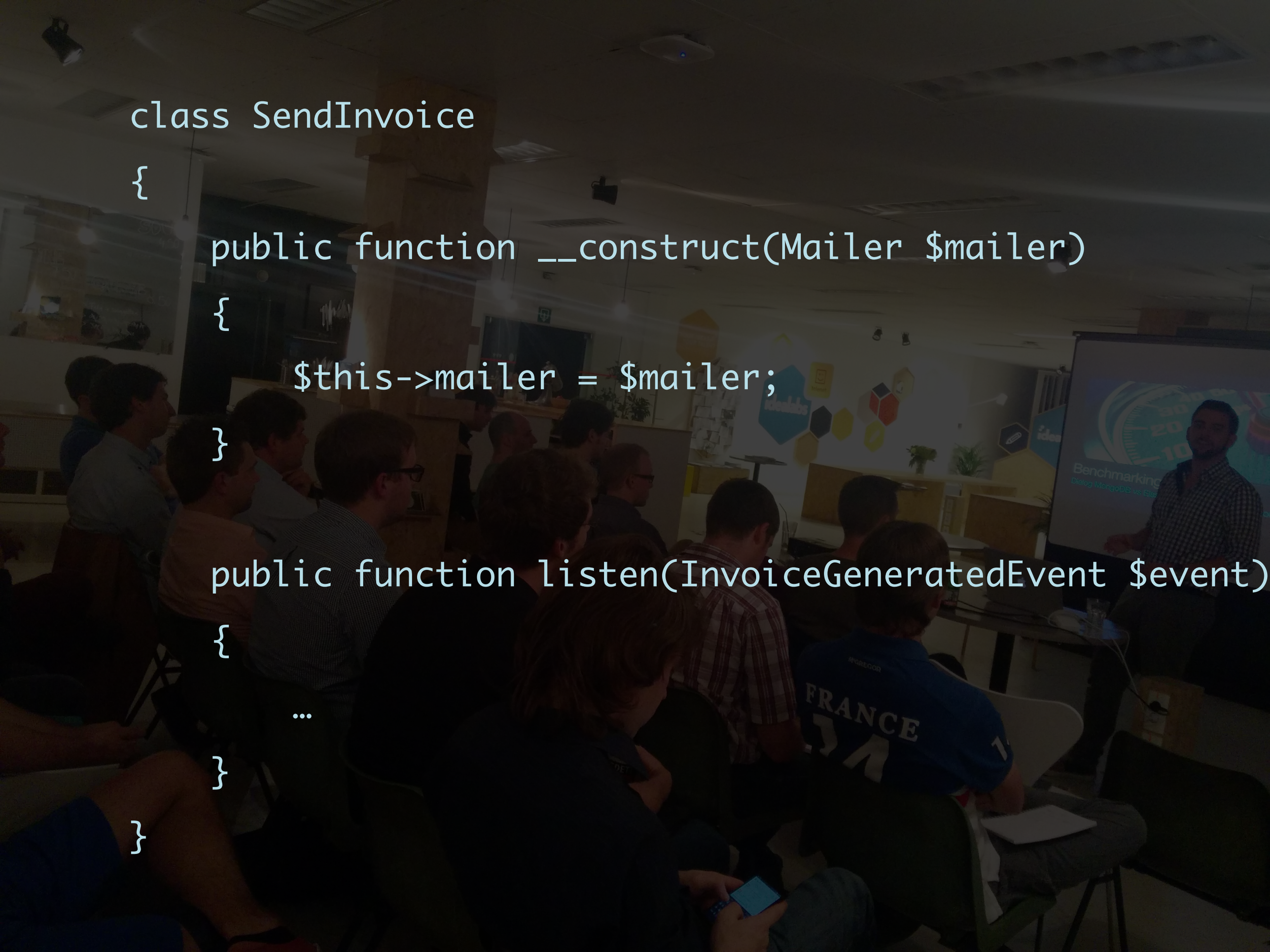
symfony/dependency-injection

aura/di

…

container-interop/container-interop

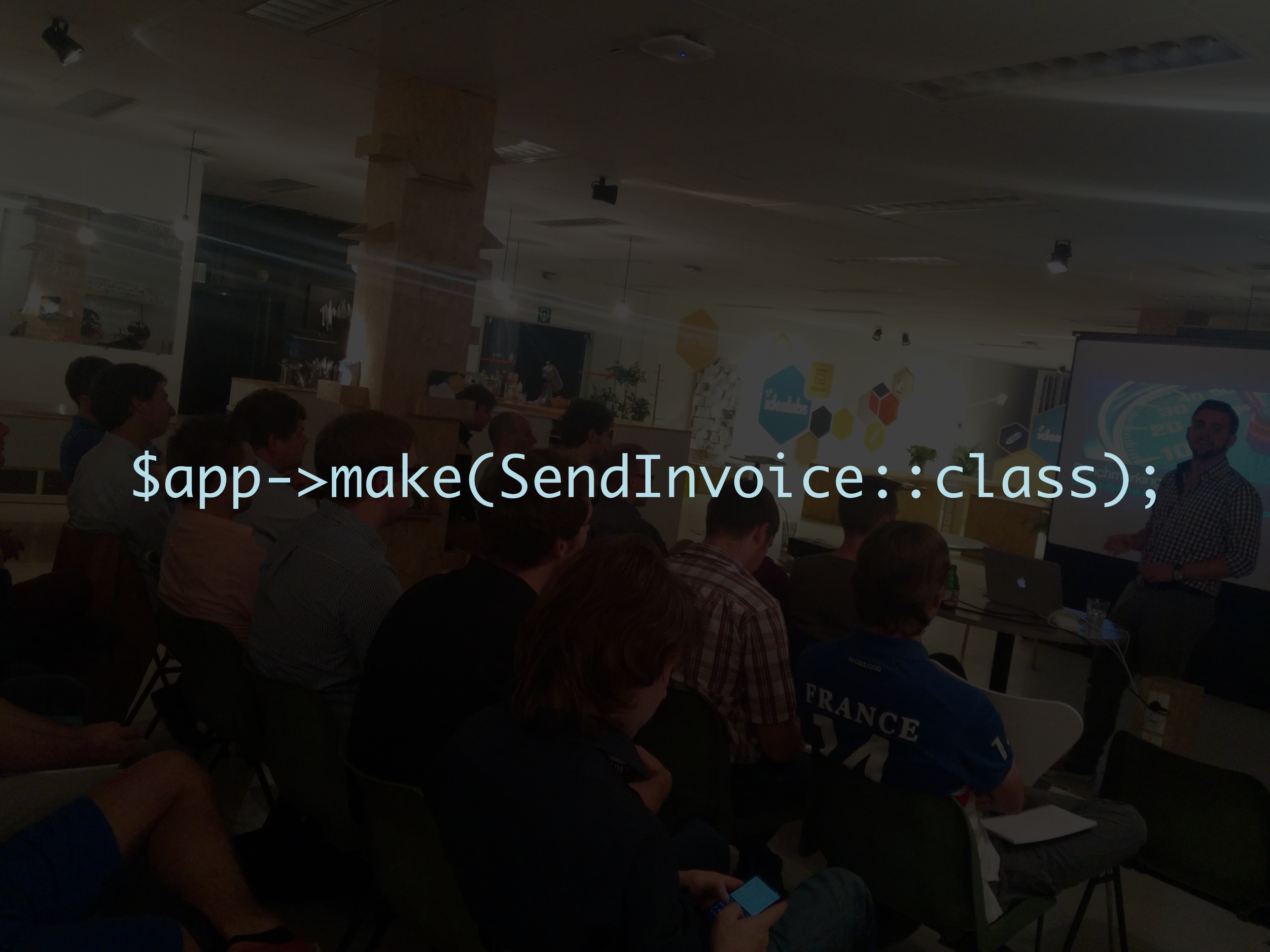
Automatic constructor injection



```
class SendInvoice
{
    public function __construct(Mailer $mailer)
    {
        $this->mailer = $mailer;
    }

    public function listen(InvoiceGeneratedEvent $event)
    {
        …
    }
}
```

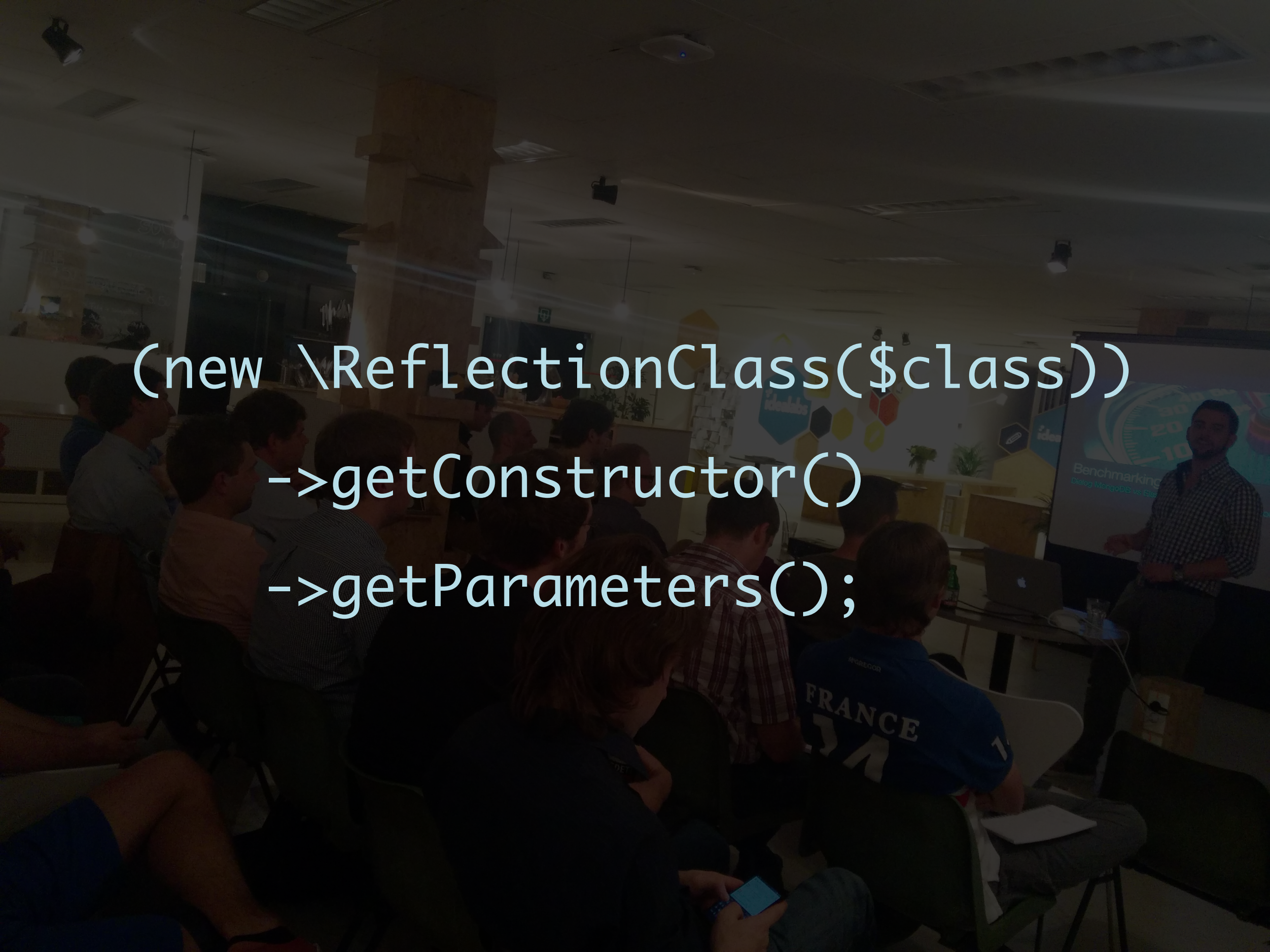
CONSTRUCTOR INJECTION

Constructor injection


$app->make(SendInvoice::class);

CONSTRUCTOR INJECTION

Reflection



```
(new \ReflectionClass($class))
    ->getConstructor()
    ->getParameters();
```

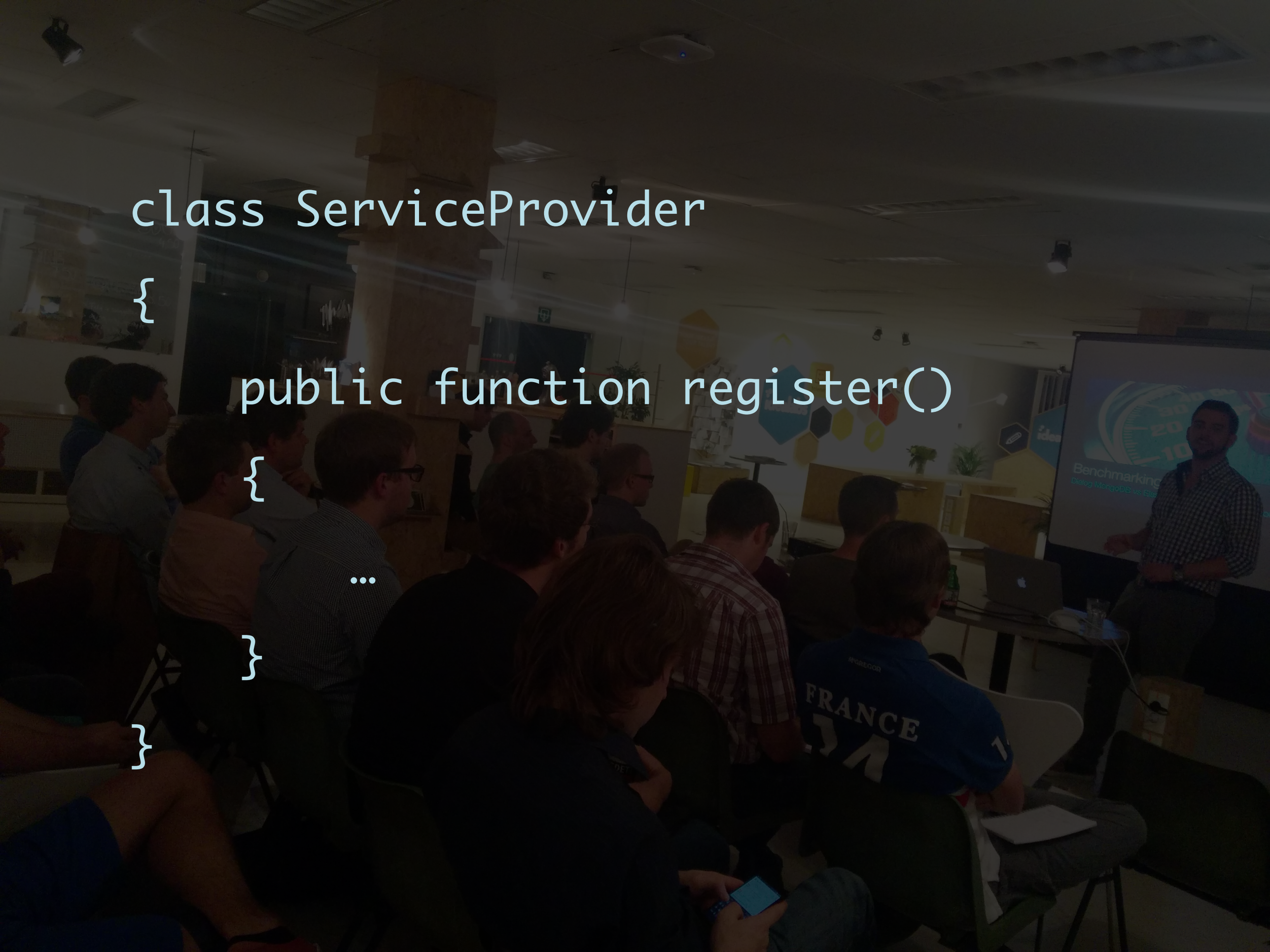
CONSTRUCTOR INJECTION

Recursively

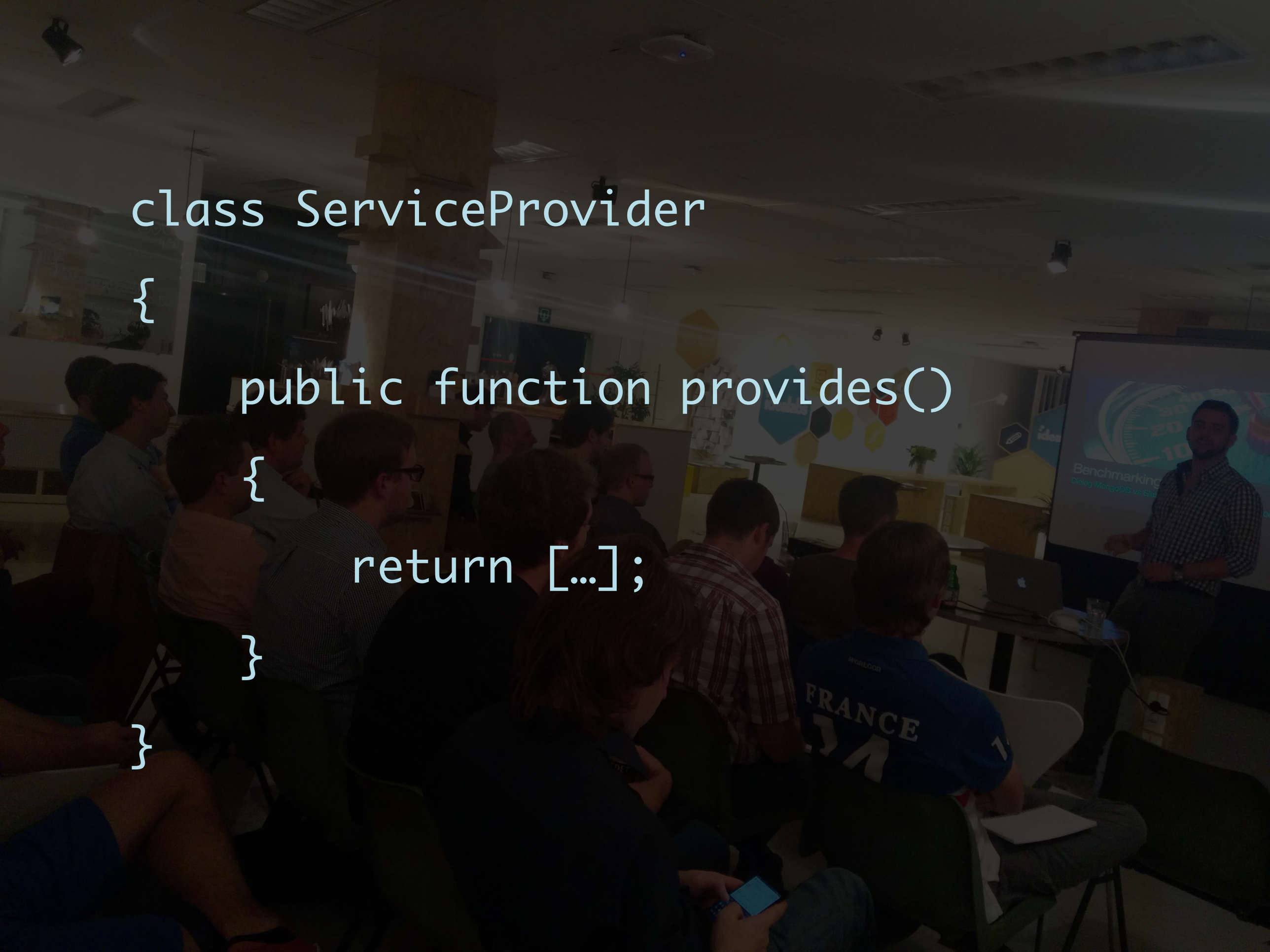
Helping the container

REGISTRATION

Service Providers


class ServiceProvider
{
public function register()
...
}
}
Benchmarking
FRANCE

```
class ServiceProvider
{
    public function boot(Mailer $mailer)
    {
        ...
    }
}
```



```
class ServiceProvider
{
    public function provides()
    {
        return […];
    }
}
```

REGISTRATION

Register bindings in the register method

REGISTRATION - BINDING

```
$this->app->bind(
    MailerContract::class,
    ConcreteMailer::class
);
```

REGISTRATION - BINDING

```
$this->app->bind(Mailer::class, function (
    ...
});
```

REGISTRATION - BINDING

```
$this->app->singleton(Mailer::class, funct
    ...
});
```

REGISTRATION - BINDING

```
$this->app->alias(
        Mailer::class,
        'mailer'
    );
```

REGISTRATION

Bind everything with its class name

REGISTRATION

Bind everything with its interface

REGISTRATION

Unless you bind multiple versions

REGISTRATION

```
$this->app->bind('fs.photos', function ()
$this->app->bind('fs.files', function () {
```

REGISTRATION - CONTEXTUAL BINDING

Contextual binding

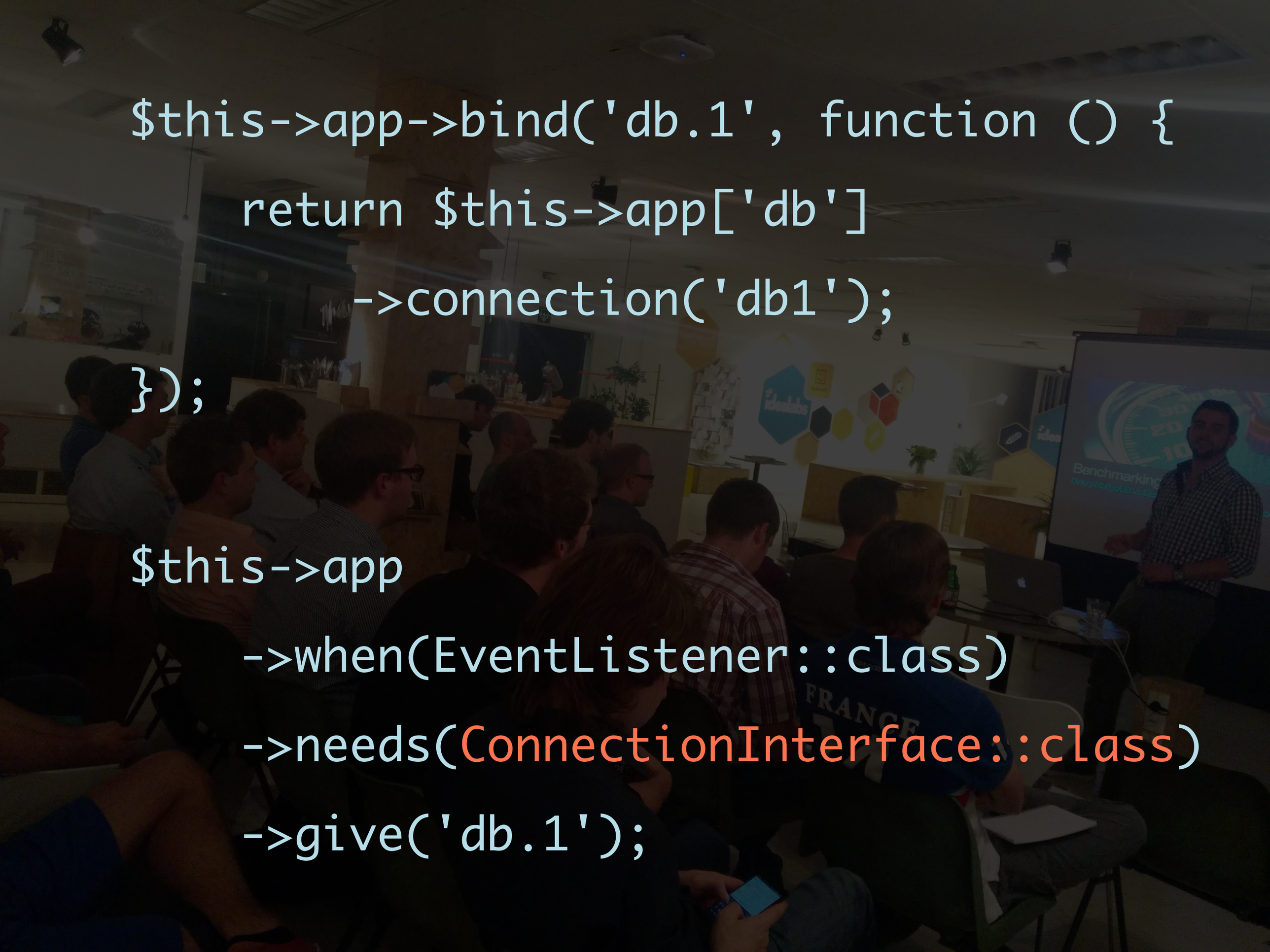
REGISTRATION - CONTEXTUAL BINDING

```
$this->app
    ->when(PhotosController::class)
    ->needs(FilesystemInterface::class)
    ->give('fs.photos');
```

REGISTRATION - CONTEXTUAL BINDING

No more

```
DB::connection('db1')->table($table)->…;
```



```php
$this->app->bind('db.1', function () {
    return $this->app['db']
        ->connection('db1');
});
```

```php
$this->app
    ->when(EventListener::class)
    ->needs(ConnectionInterface::class)
    ->give('db.1');
```

REGISTRATION - CONTAINER EVENTS

Inflection

REGISTRATION - CONTAINER EVENTS

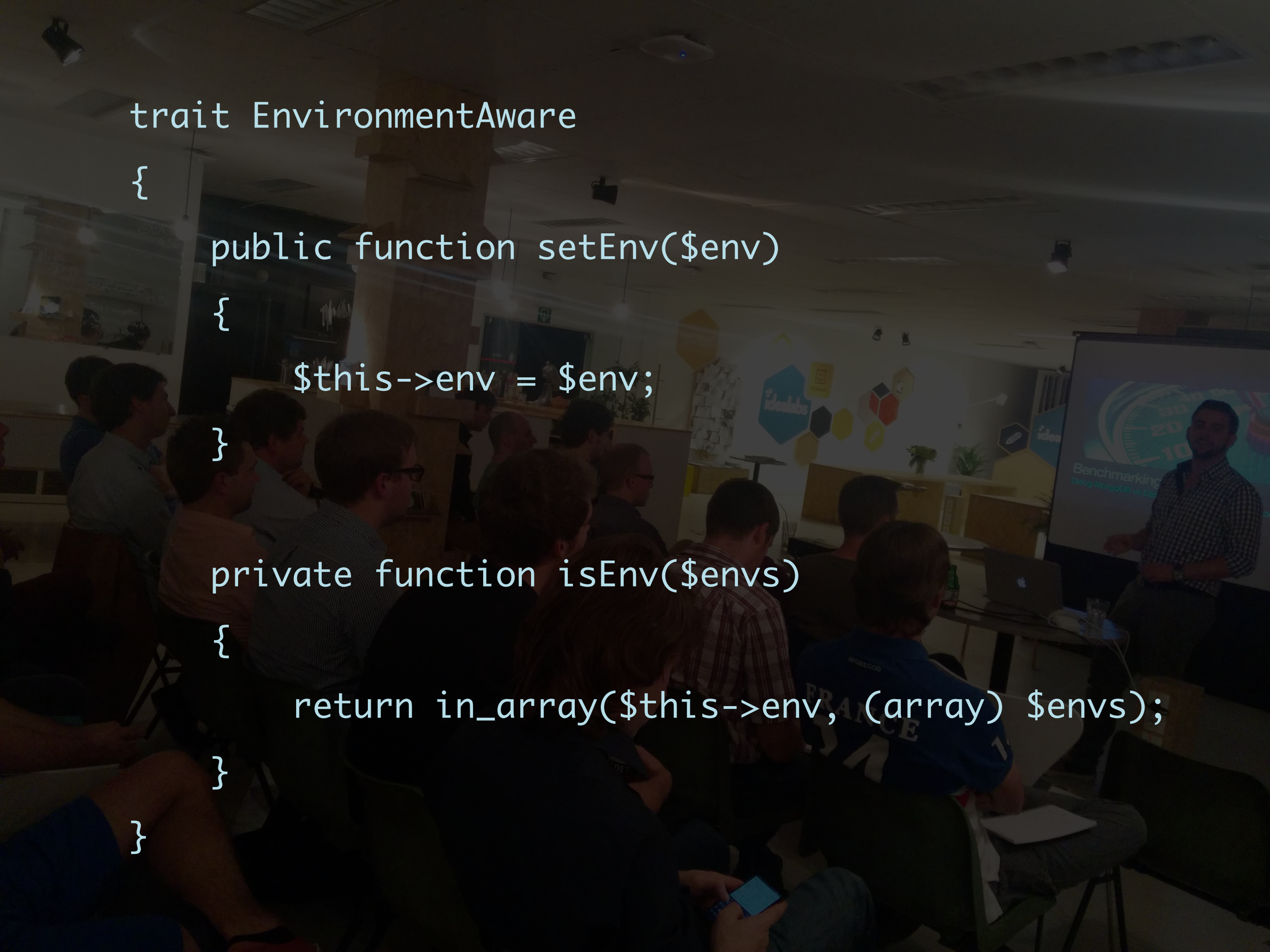
Inflection (aka “container events”)

REGISTRATION - CONTAINER EVENTS

```
$this->app->resolving(function () {});
```

```
use App\Contracts\EnvironmentAware;

$this->app->resolving(
    function (EnvironmentAware $object) {
        $object->setEnv(
            $this->app->environment()
        );
    }
);
```

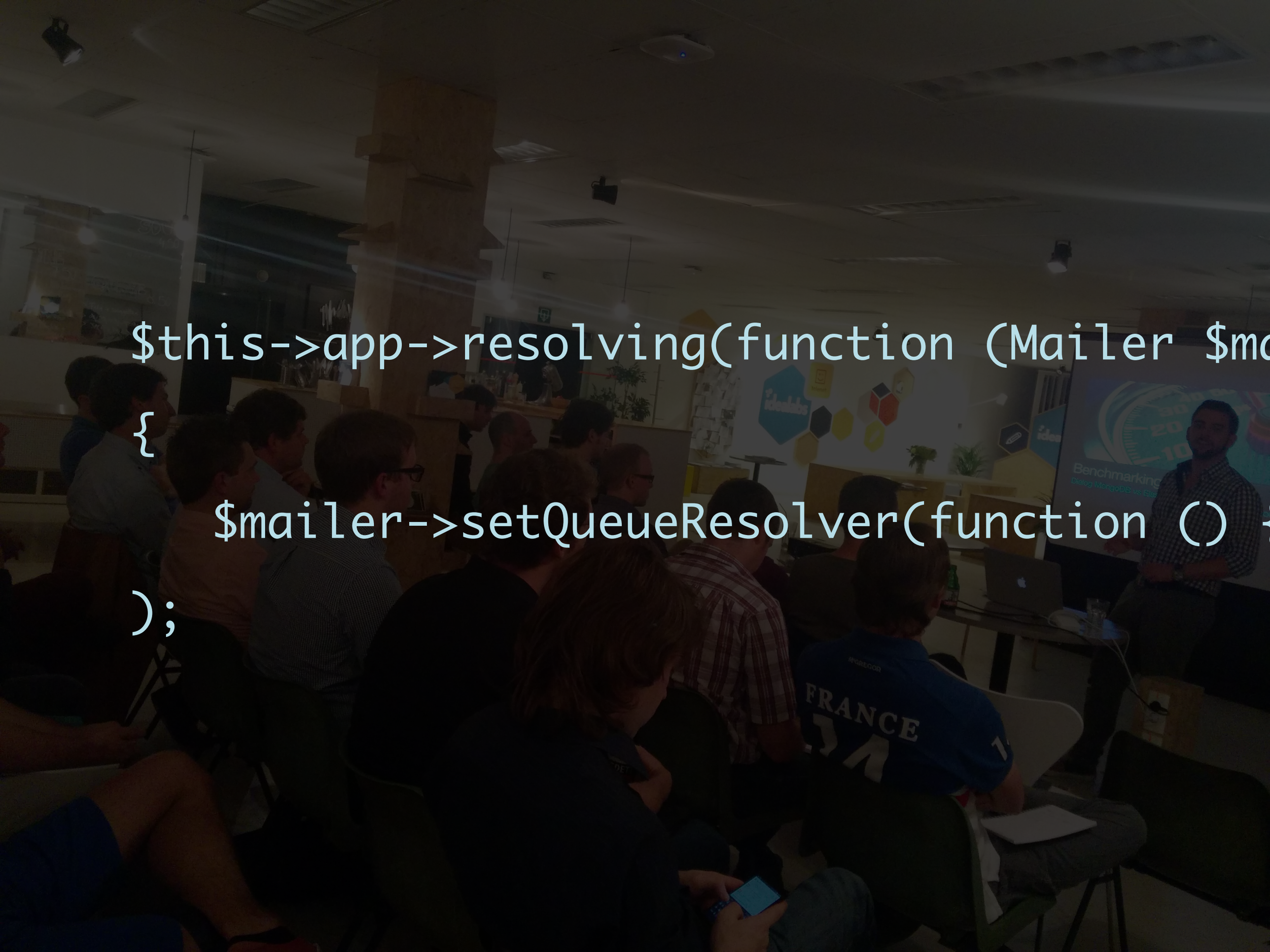
```
interface EnvironmentAware

{

    public function setEnv($env);

}
```

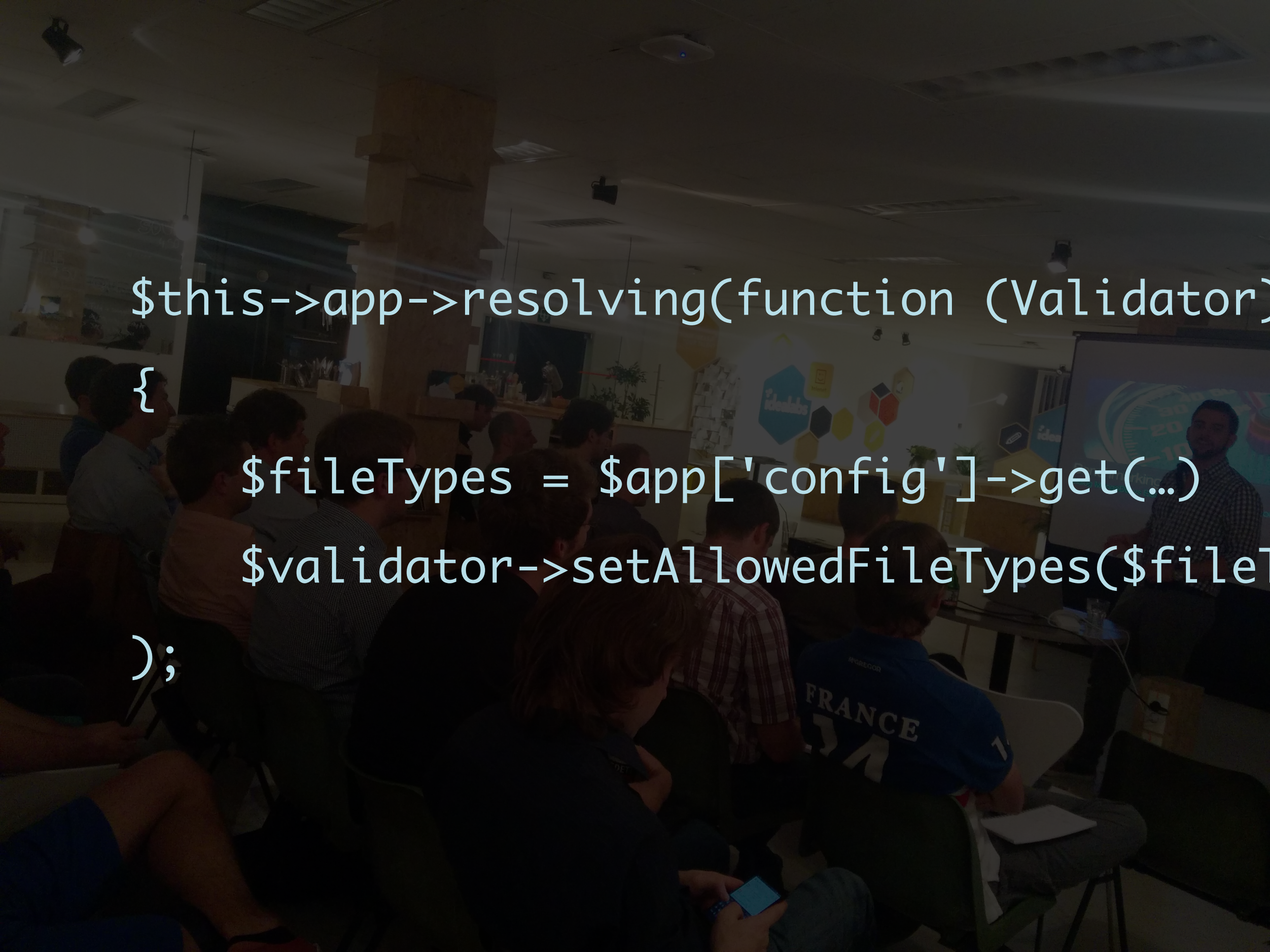


```
trait EnvironmentAware
{
    public function setEnv($env)
    {
        $this->env = $env;
    }

    private function isEnv($envs)
    {
        return in_array($this->env, (array) $envs);
    }
}
```

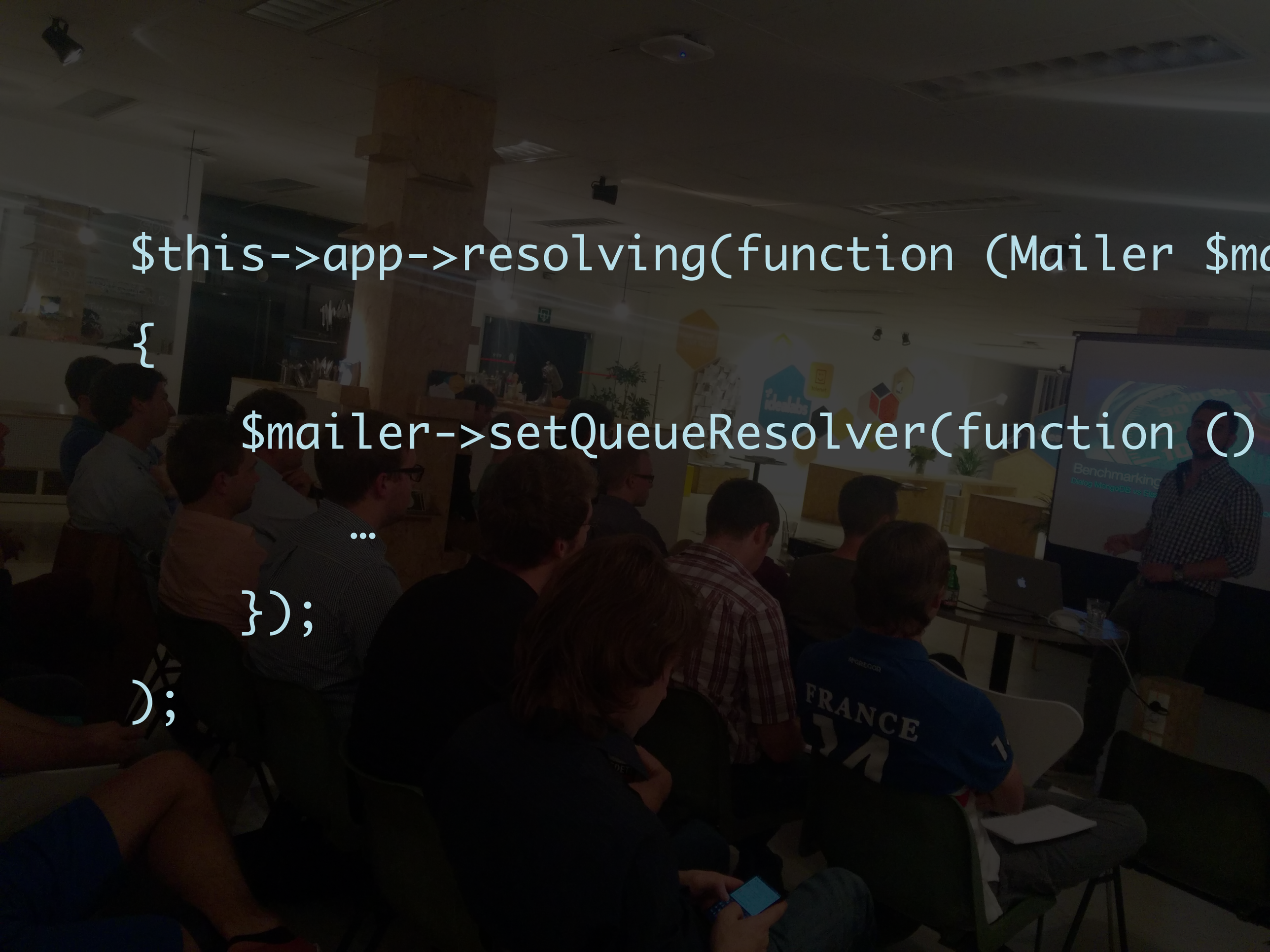
REGISTRATION - CONTAINER EVENTS

Method injection


$this->app->resolving(function (Mailer $ma
{
$mailer->setQueueResolver(function () {
});



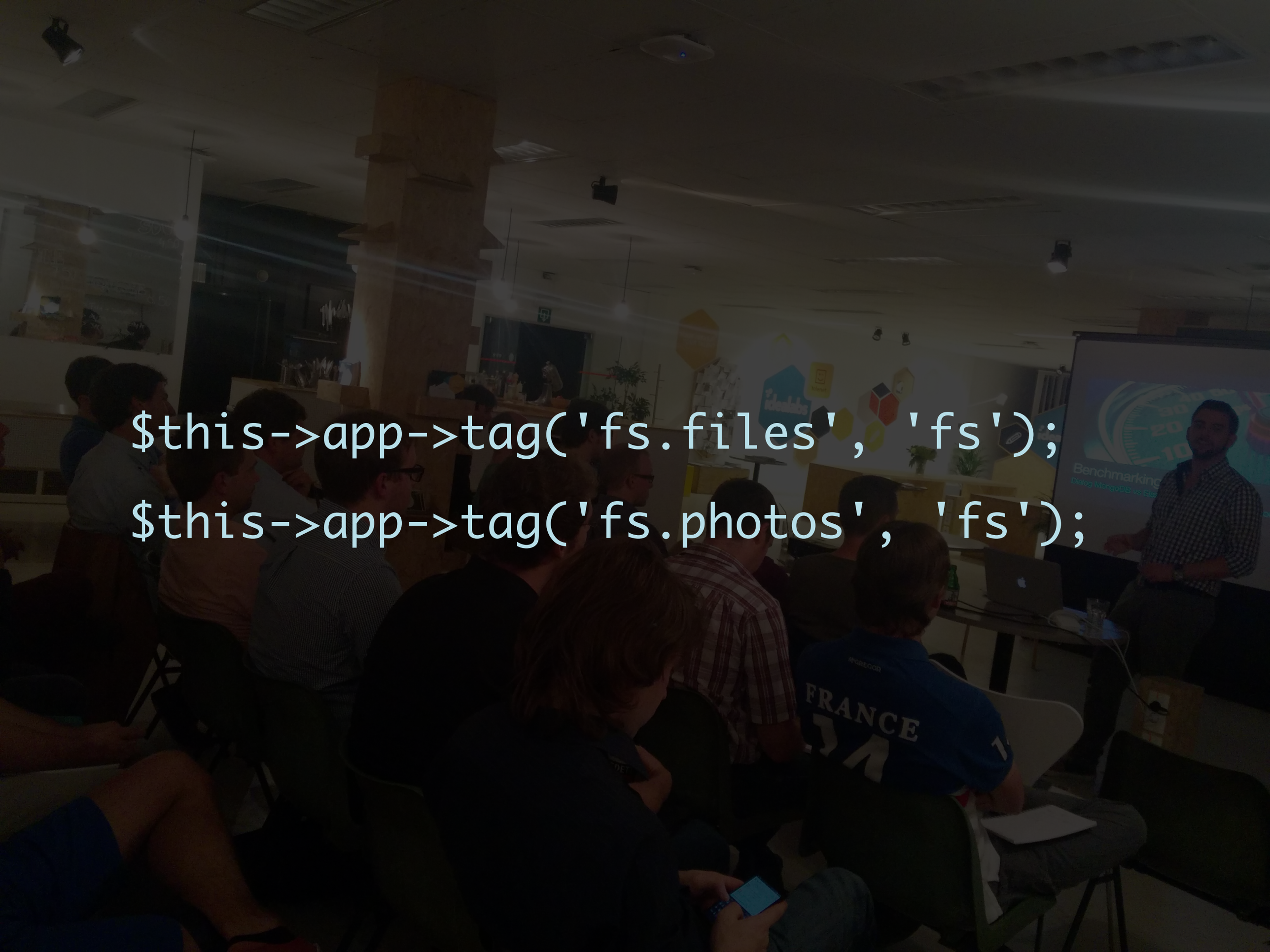
```
$this->app->resolving(function (Validator)
{
    $fileTypes = $app['config']->get(…)
    $validator->setAllowedFileTypes($fileT
);
```



```
$this->app->resolving(function (Mailer $ma
{
    $mailer->setQueueResolver(function ()
        ...
    });
);
```

REGISTRATION - TAGGING

Tagging


$this->app->tag('fs.files', 'fs');
$this->app->tag('fs.photos', 'fs');

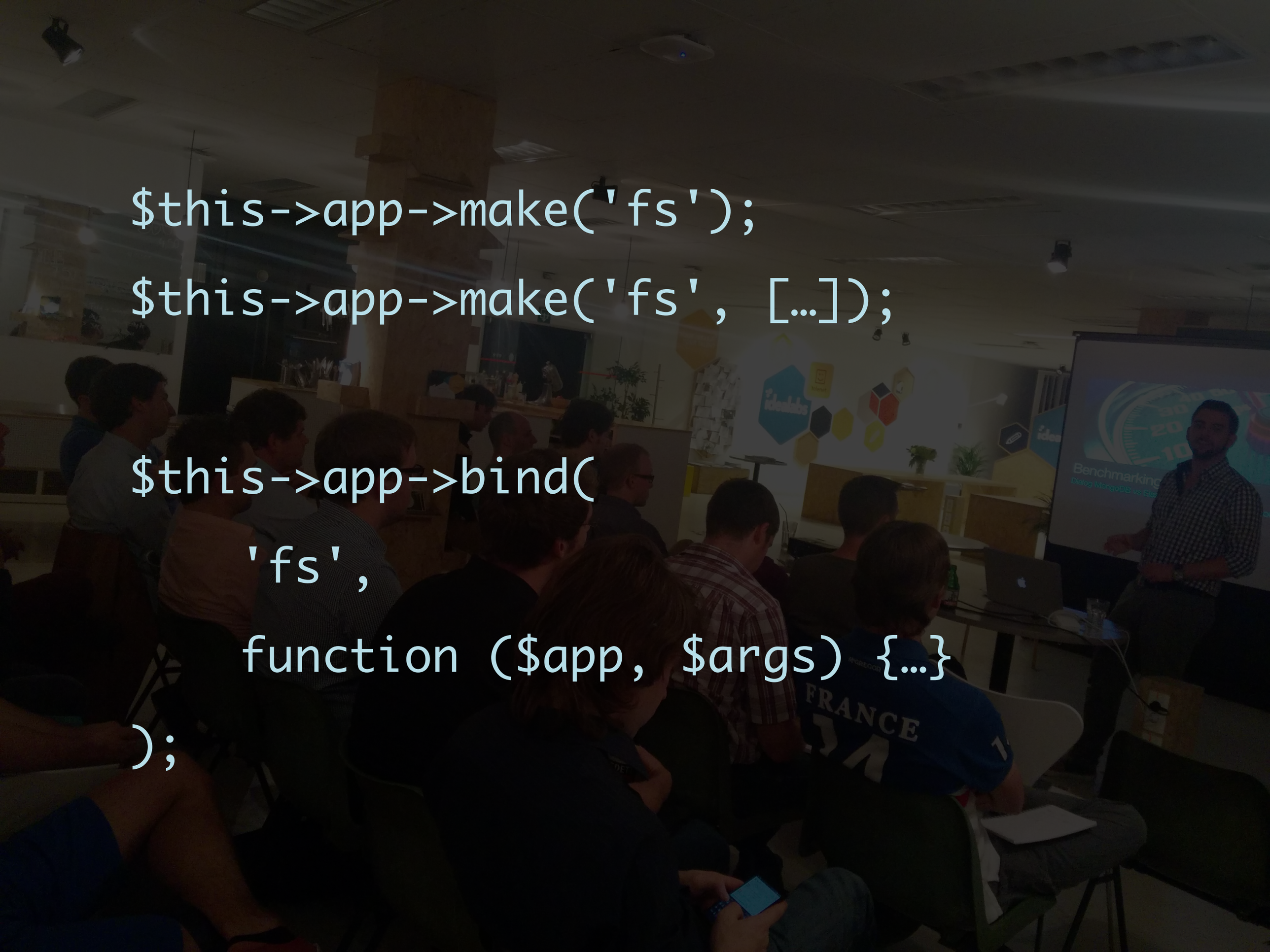
```
$this->app->tagged('fs');
```

Using the container

CALLING THE CONTAINER

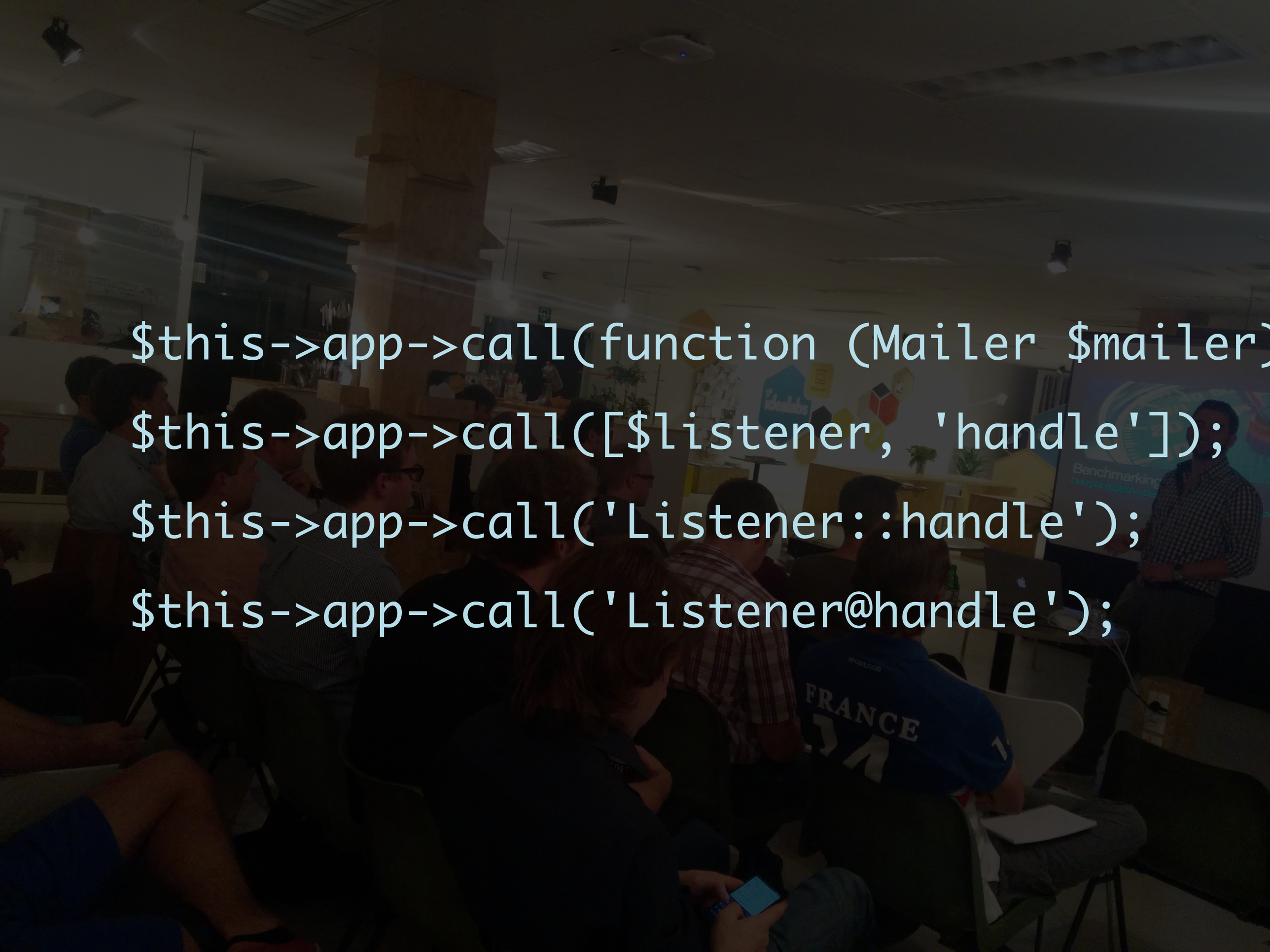
Preferably only for

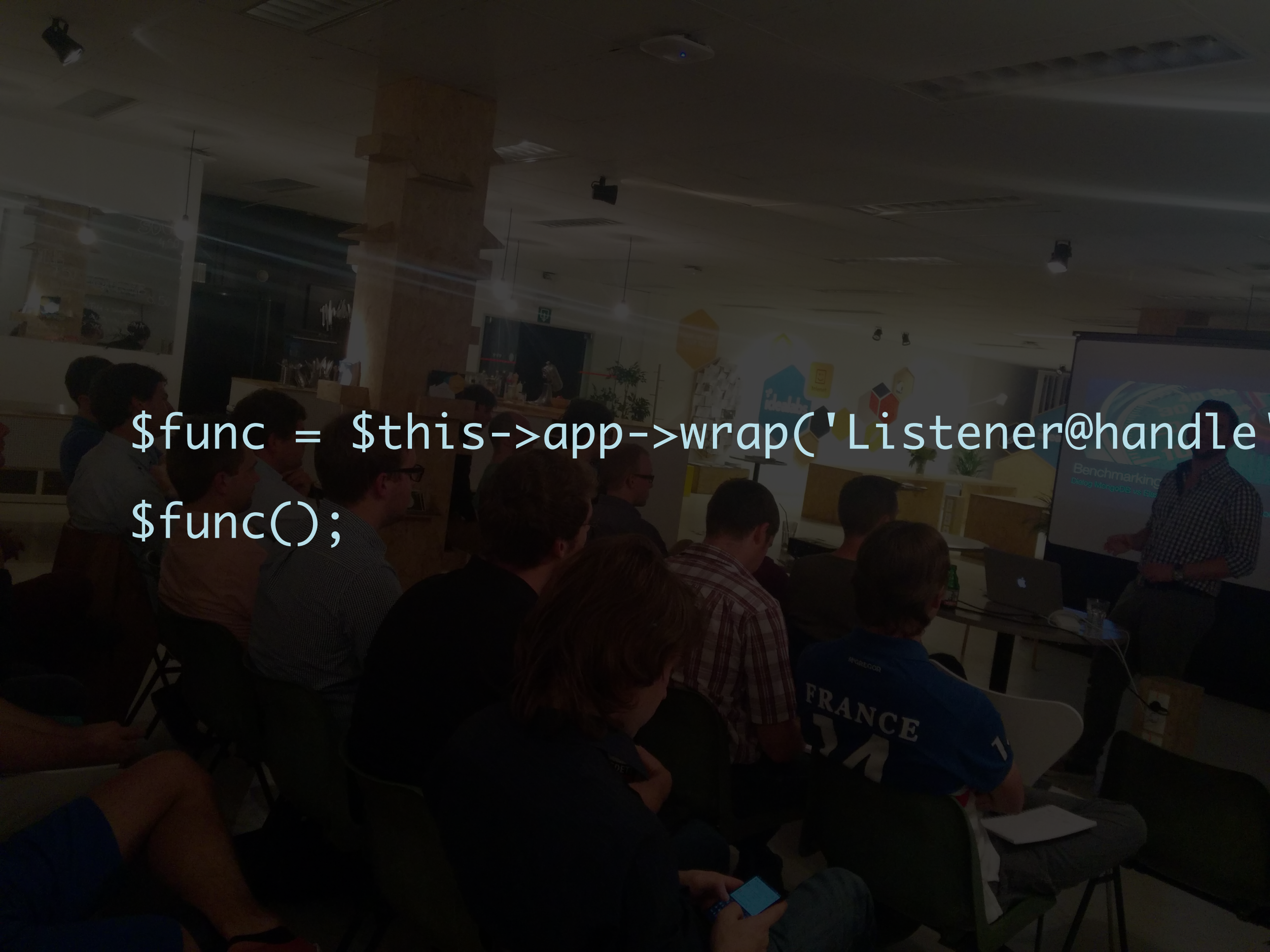
- Factories
- Event Listeners
- Controllers
- Command Handlers



```
$this->app->make('fs');
$this->app->make('fs', […]);
```

```
$this->app->bind(
    'fs',
    function ($app, $args) {…}
);
```


$this->app->call(function (Mailer $mailer)
$this->app->call([$listener, 'handle']);
$this->app->call('Listener::handle');
$this->app->call('Listener@handle');


$func = $this->app->wrap('Listener@handle
$func();
FRANCE

```
$this->app->tagged('fs');
```

Bonus

BONUS - SOFT DEPENDENCIES

Solving soft dependencies

BONUS - SOFT DEPENDENCIES

```
$func = function () {
    return $this->app->make('queue');
}

$mailer->setQueueResolver($func);
```

public **function setUserResolver**(Closure $callback) Request.php 926

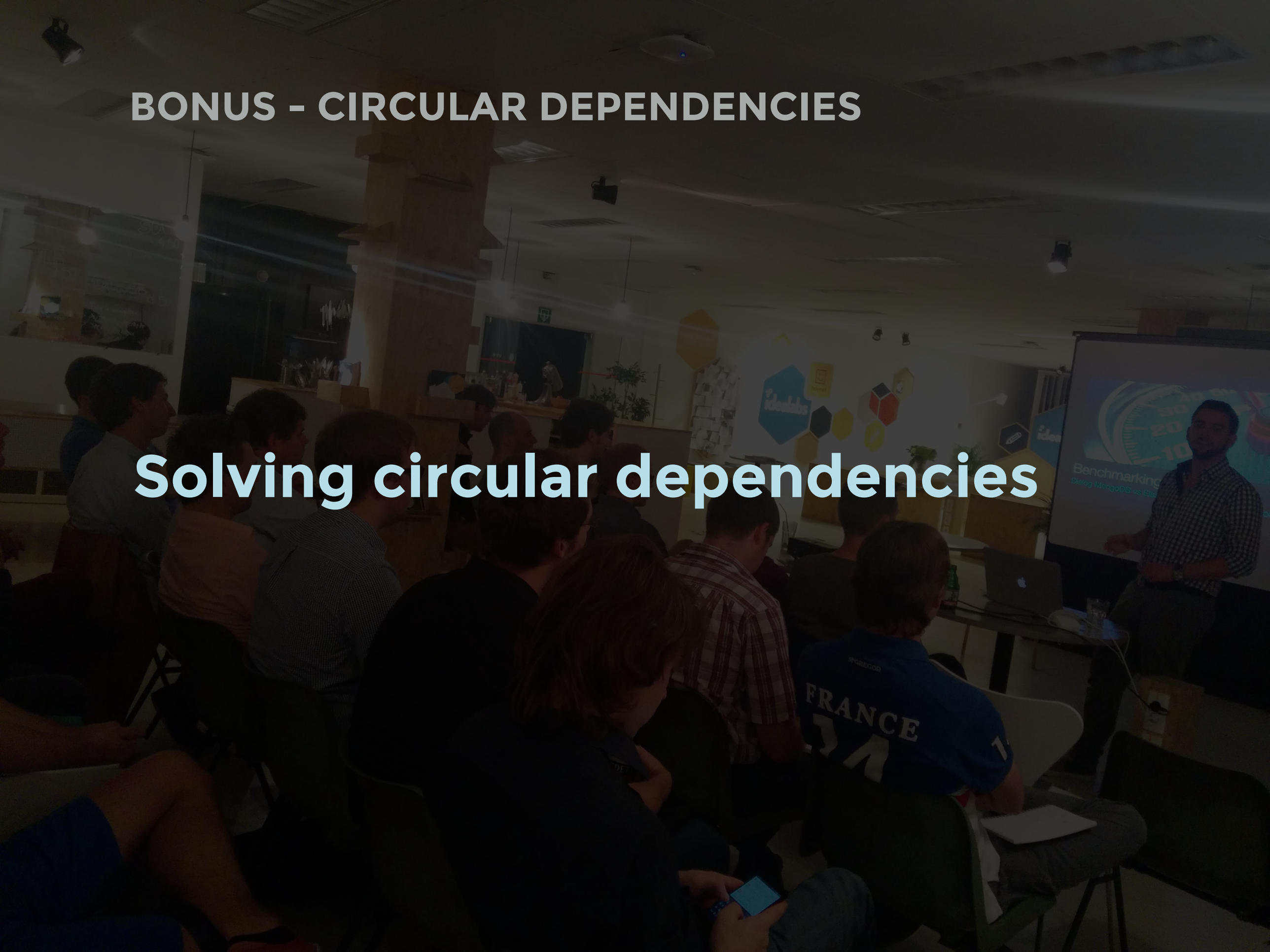
public **function setRouteResolver**(Closure $callback) Request.php 951

public **function setSessionResolver**(callable $sessionResolver) UrlGenerator.php 735

public **function setQueueResolver**(callable $resolver) Dispatcher.php 497

public static **function setConnectionResolver**(Resolver $resolver) Model.php 3299

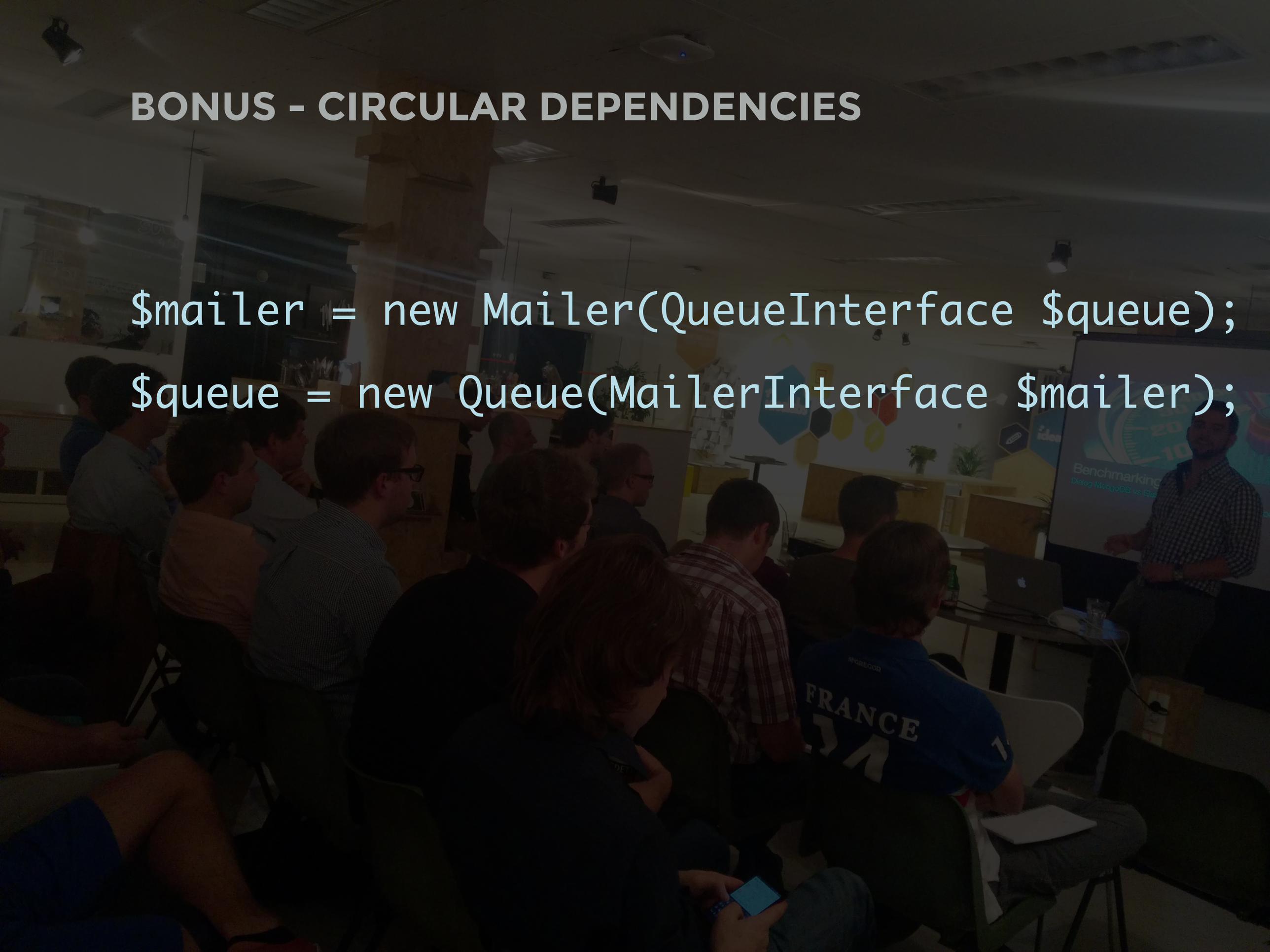
BONUS - CIRCULAR DEPENDENCIES



Solving circular dependencies

BONUS - CIRCULAR DEPENDENCIES

```
$mailer = new Mailer(QueueInterface $queue);
$queue = new Queue(MailerInterface $mailer);
```



BONUS - CIRCULAR DEPENDENCIES

```
$mailer = new Mailer();
$queue = new Queue(MailerInterface $mailer);
$mailer->setQueueResolver(function () use ($q
    return $queue;
});
```

BONUS - NOT SUPPORTED

BONUS - NOT SUPPORTED

Contextual binding for methods

BONUS - NOT SUPPORTED

https://github.com/laravel/framework/pull/12183

BONUS - CIRCULAR DEPENDENCIES

```
$this->app->when('Foo@bar')
    ->needs(QueueContract::class)
    ->give('queue.redis');
```

Reflection
Binding
Calling code
Bonus

Thank you!

https://joind.in/talk/xxxxx

#laravelcologne @hannesvdvreken

Time for questions.

#laravelcologne @hannesvdvreken

REFERENCES

- http://mwl.be