Clean Code

About us

- Thanos Korakas, in/in/tkorakas
- Thanos Nokas, in/in/thanosnokas

```
let social = [  ];

social.forEach(network => {
  console.log(`${network}/SocialNerdsGR`);
});
```

SocialNerds

- Mentoring
- Private workshops
- Consulting
- Recruitment
- (Put more marketing keywords here!!!!1)

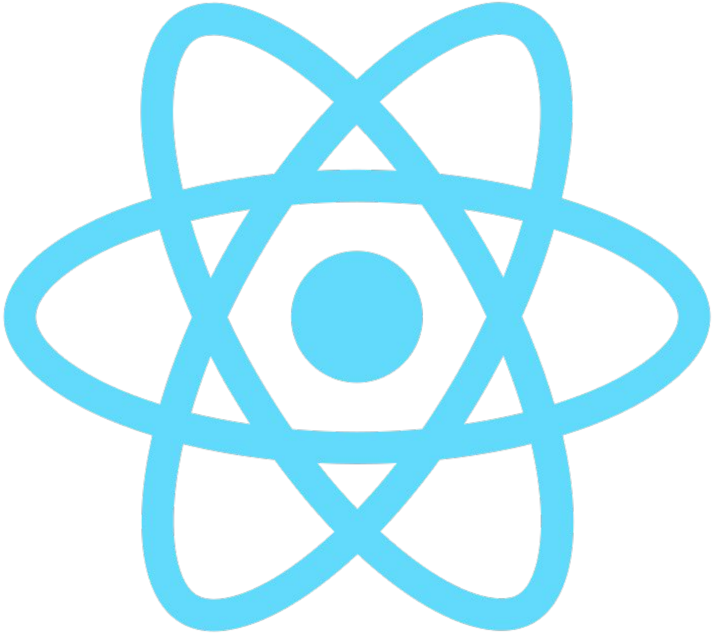
DESIGN PATTERNS
DESIGN PATTERNS EVERYWHERE
makeameme.org

Design patterns

> *“A software design pattern is a general, reusable solution to a commonly occurring problem within a given context in software design”*

- *Wikipedia*

Microservices

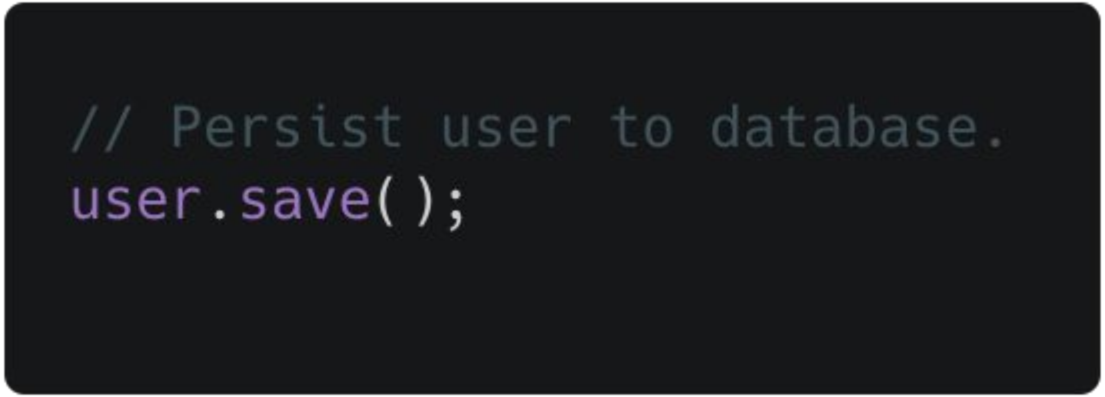


WHAT IF I TOLD YOU
THERE IS NO SUCH THING AS "LEGACY CODE"
imgflip.com

Good habits

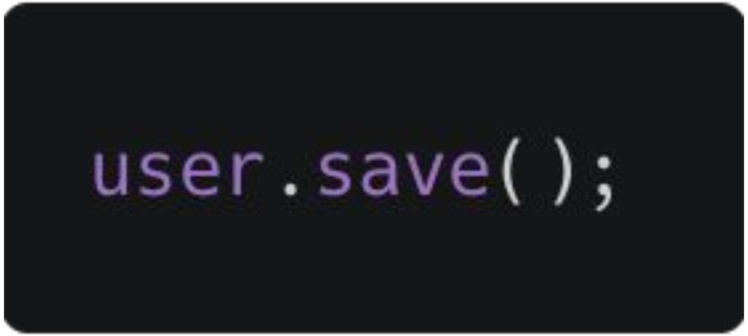
Comments

Obvious



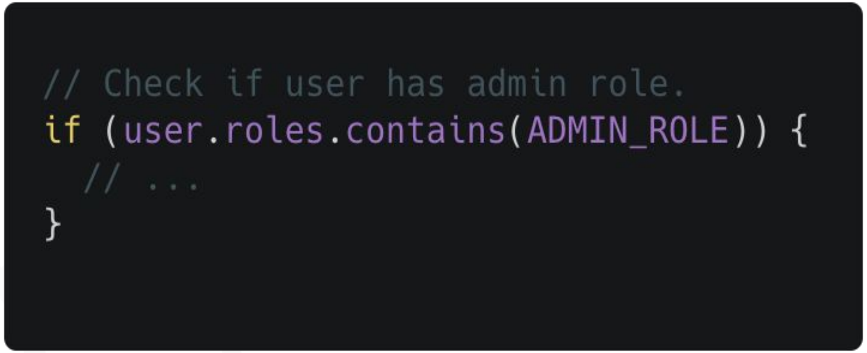
```
// Persist user to database.
user.save();
```

Obvious

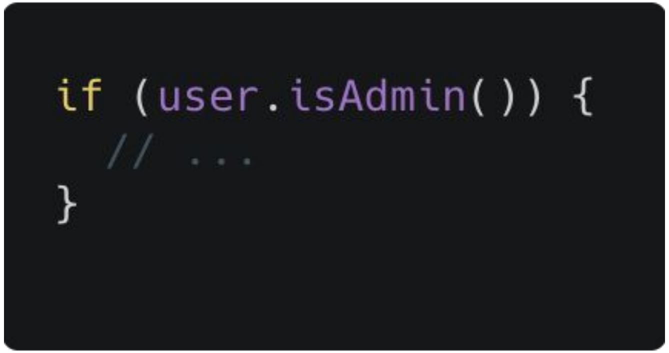


```
user.save();
```

Write code that doesn't need comments

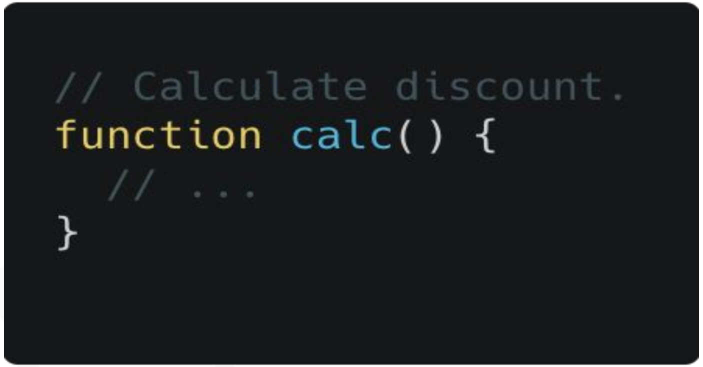


```
// Check if user has admin role.
if (user.roles.contains(ADMIN_ROLE)) {
  // ...
}
```

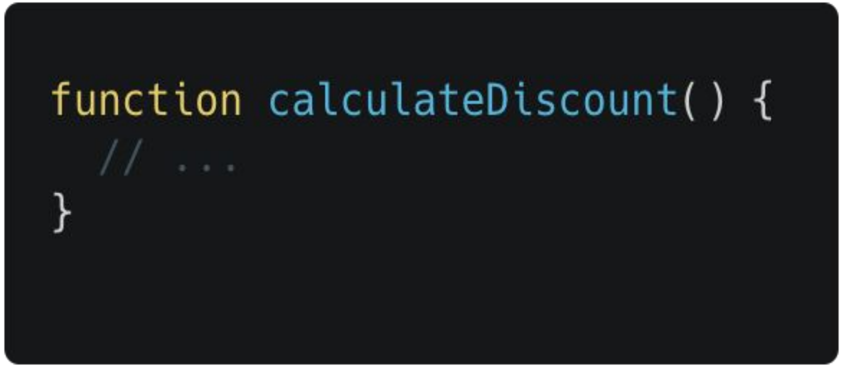


```
if (user.isAdmin()) {
  // ...
}
```

Proper naming

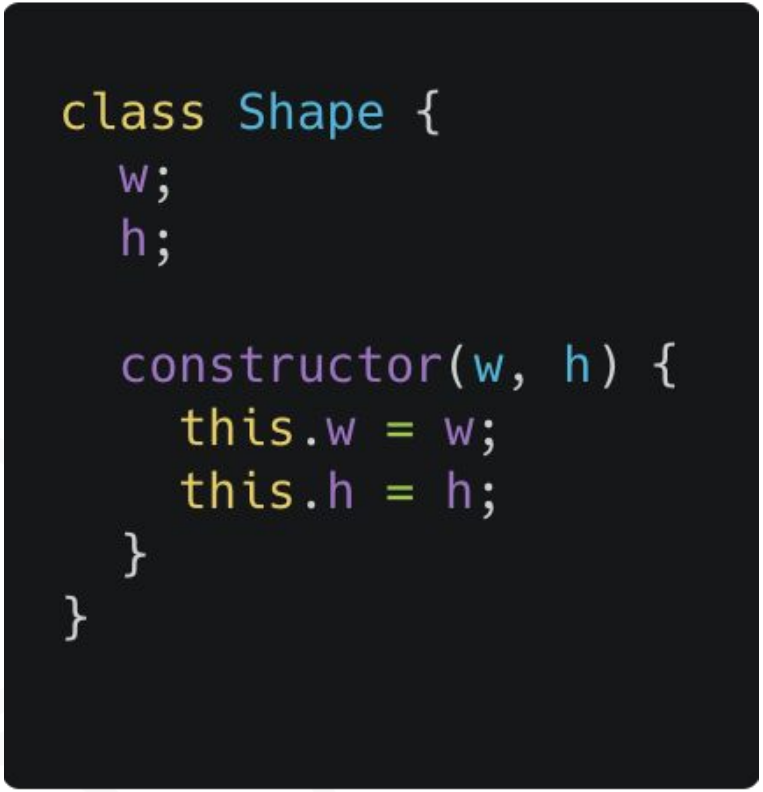


```
// Calculate discount.
function calc() {
  // ...
}
```

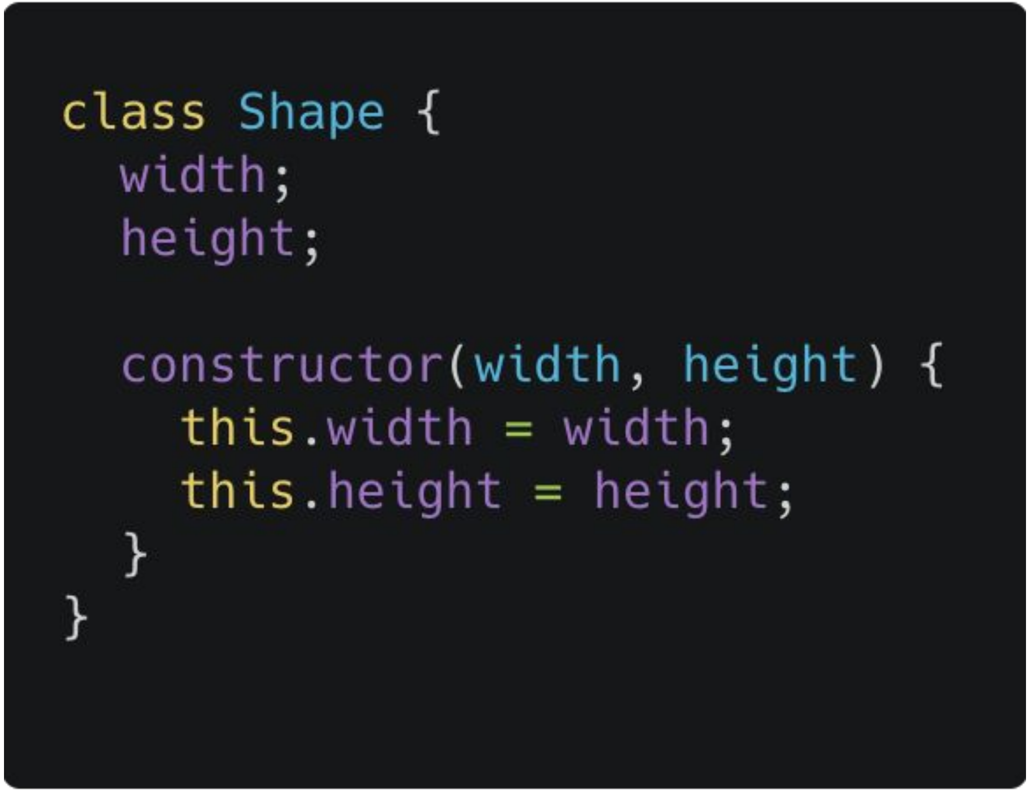


```
function calculateDiscount() {
  // ...
}
```

Proper naming

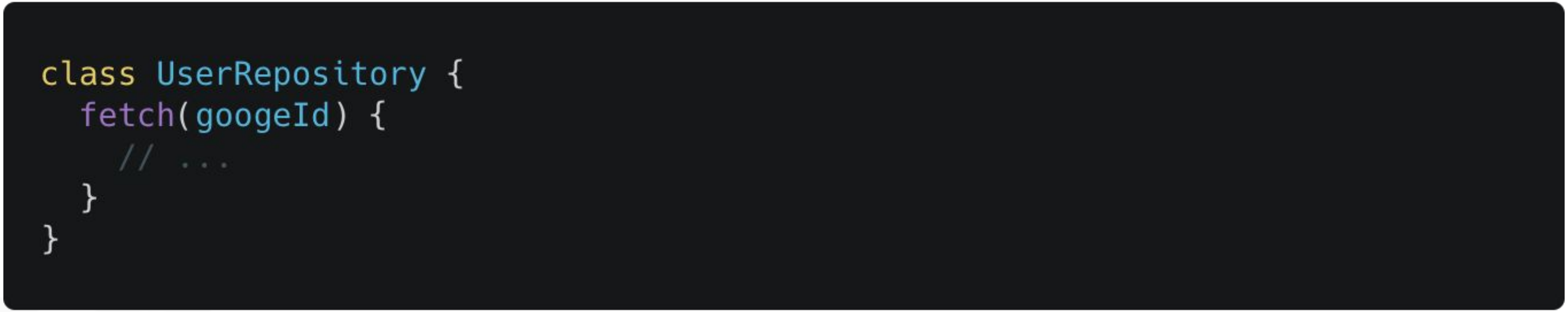


```
class Shape {
  w;
  h;

  constructor(w, h) {
    this.w = w;
    this.h = h;
  }
}
```

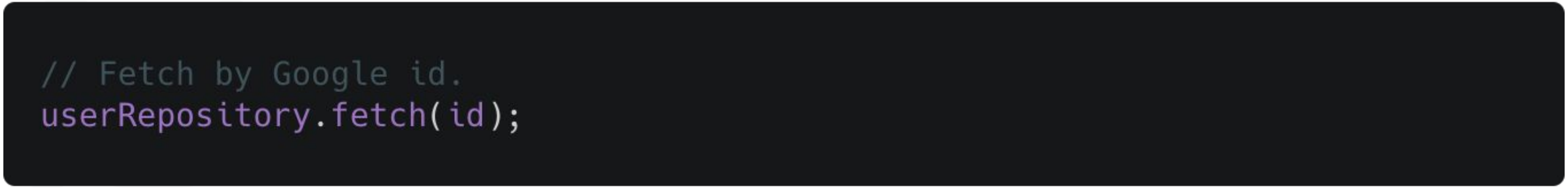


```
class Shape {
  width;
  height;

  constructor(width, height) {
    this.width = width;
    this.height = height;
  }
}
```

Proper naming

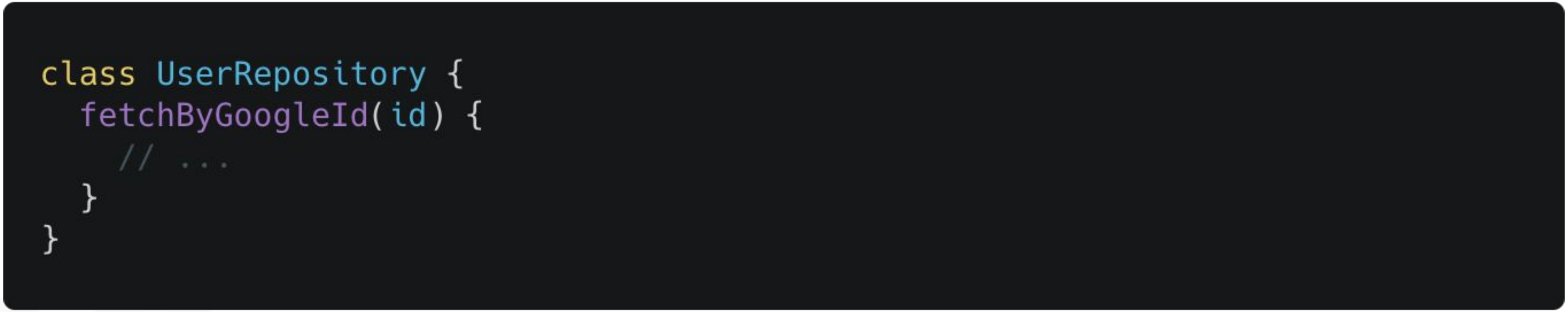


```
class UserRepository {
  fetch(googeId) {
    // ...
  }
}
```

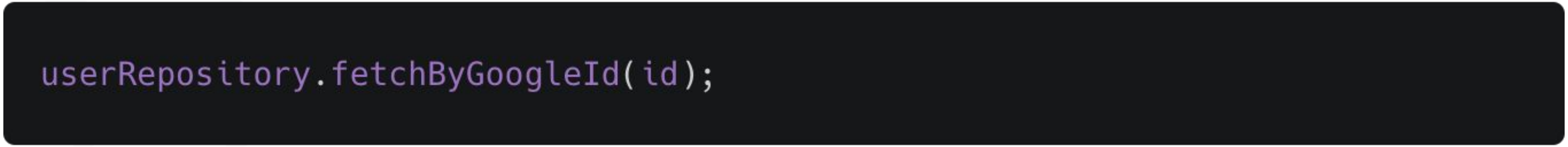


```
// Fetch by Google id.
userRepository.fetch(id);
```

Proper naming

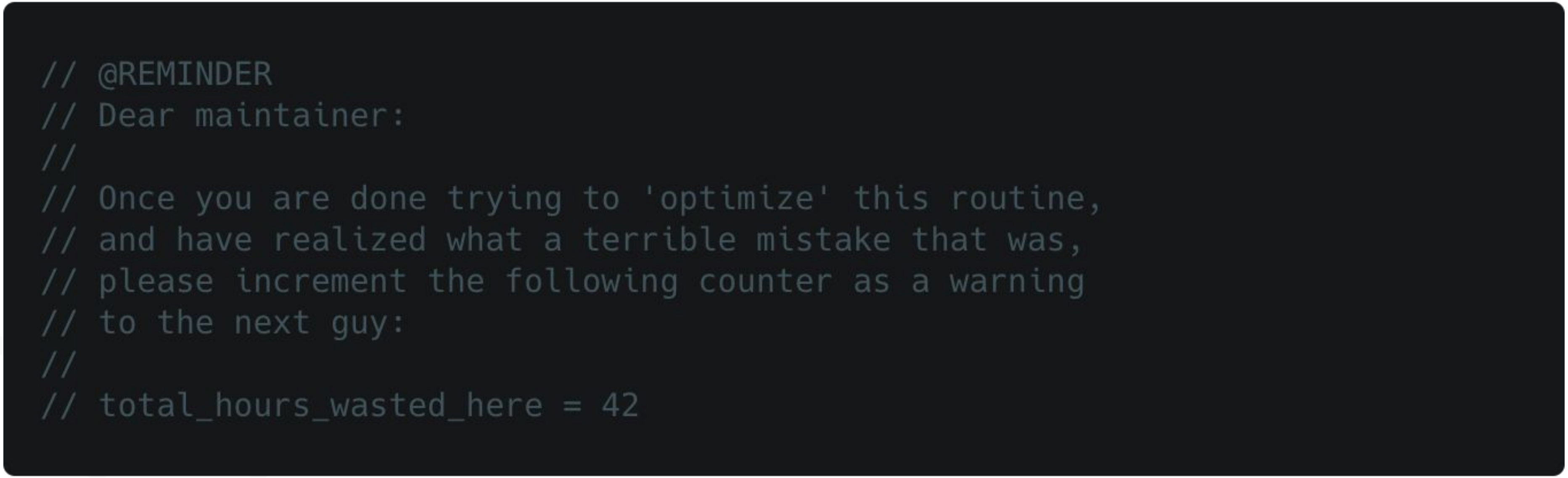


```
class UserRepository {
  fetchByGoogleId(id) {
    // ...
  }
}
```



```
userRepository.fetchByGoogleId(id);
```

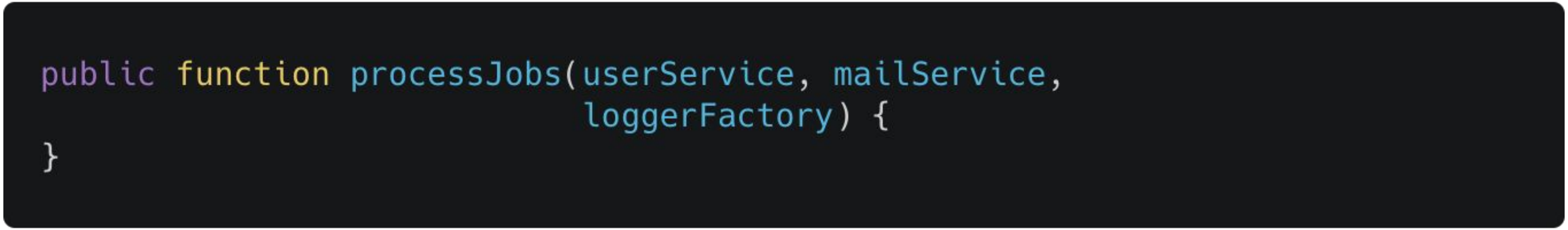
Sometimes comments are necessary



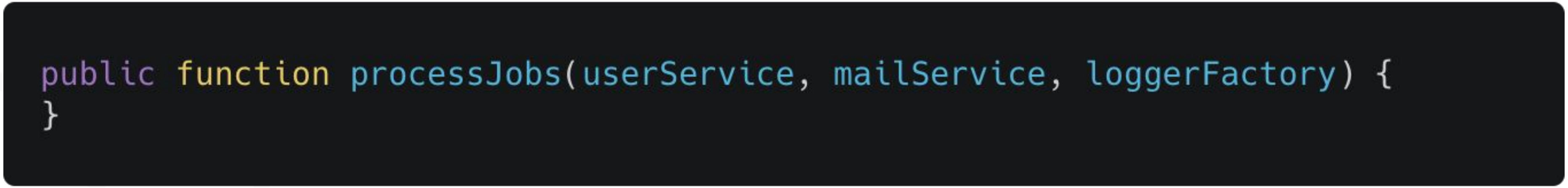
```
// @REMINDER
// Dear maintainer:
//
// Once you are done trying to 'optimize' this routine,
// and have realized what a terrible mistake that was,
// please increment the following counter as a warning
// to the next guy:
//
// total_hours_wasted_here = 42
```

Coding standards

Arguments

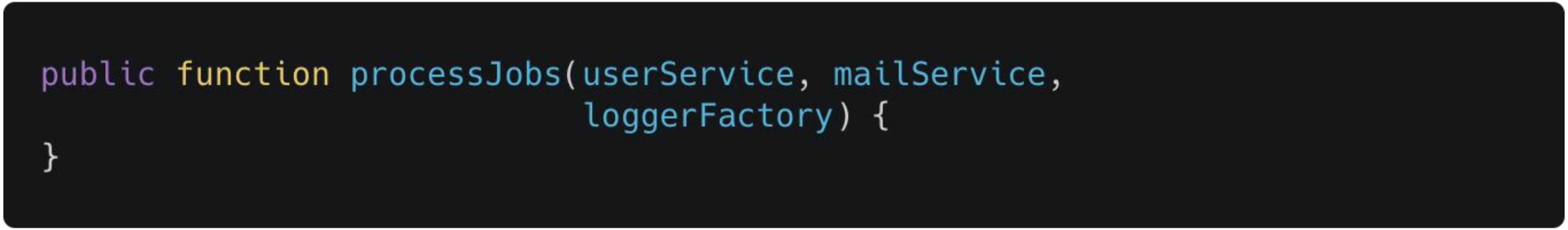


```
public function processJobs(userService, mailService,
                            loggerFactory) {
}
```

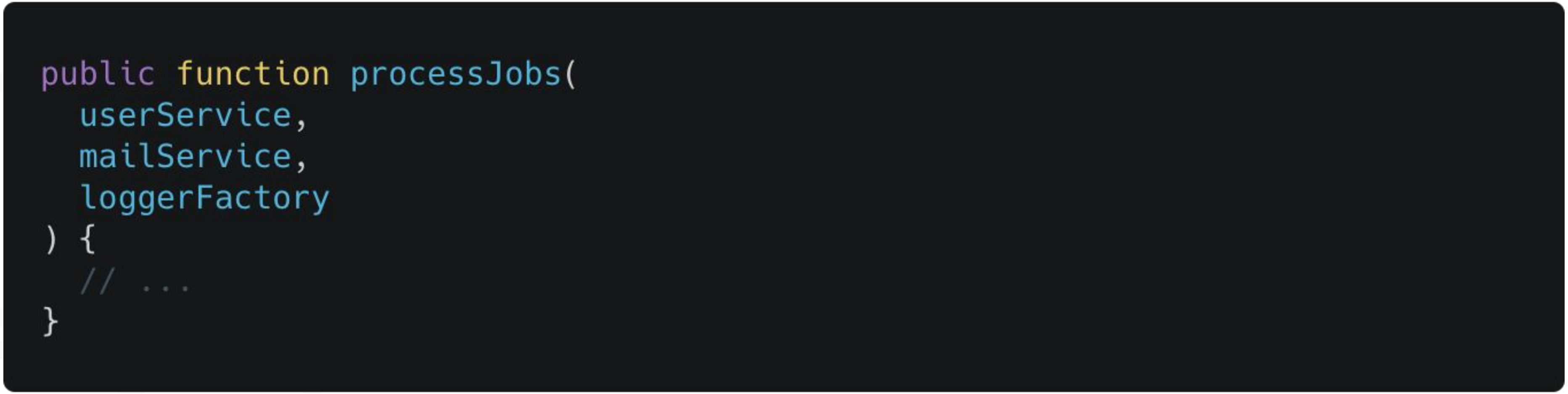


```
public function processJobs(userService, mailService, loggerFactory) {
}
```

Arguments

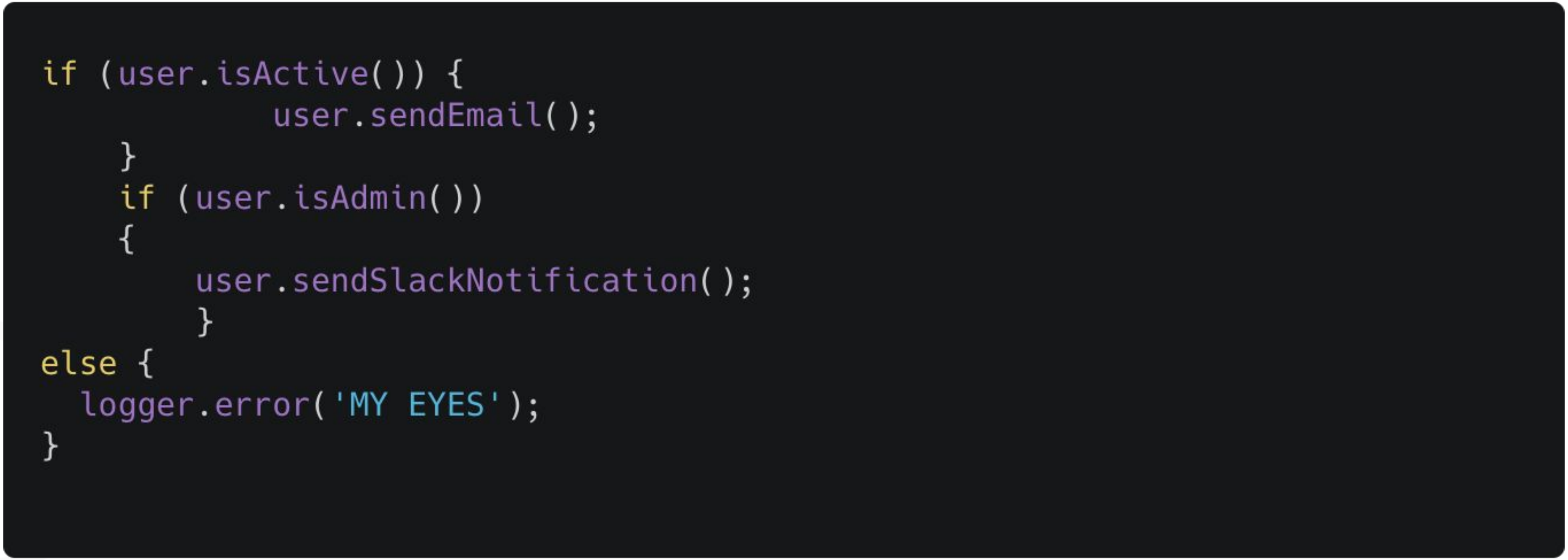


```
public function processJobs(userService, mailService,
                            loggerFactory) {
}
```



```
public function processJobs(
  userService,
  mailService,
  loggerFactory
) {
  // ...
}
```

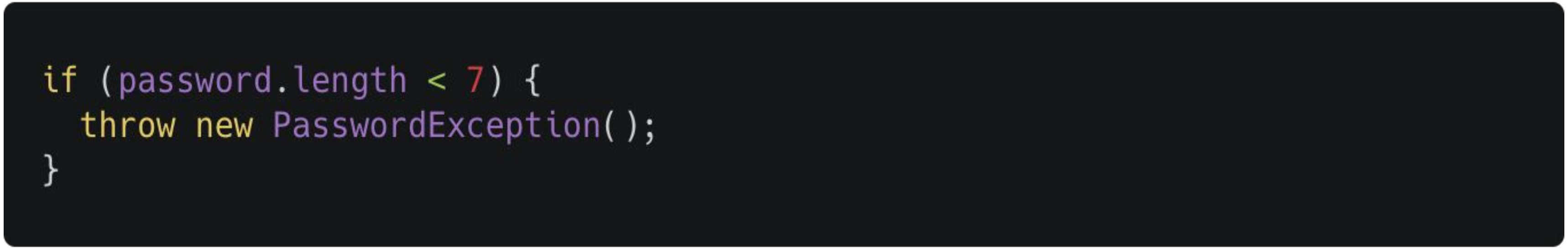
Indentation



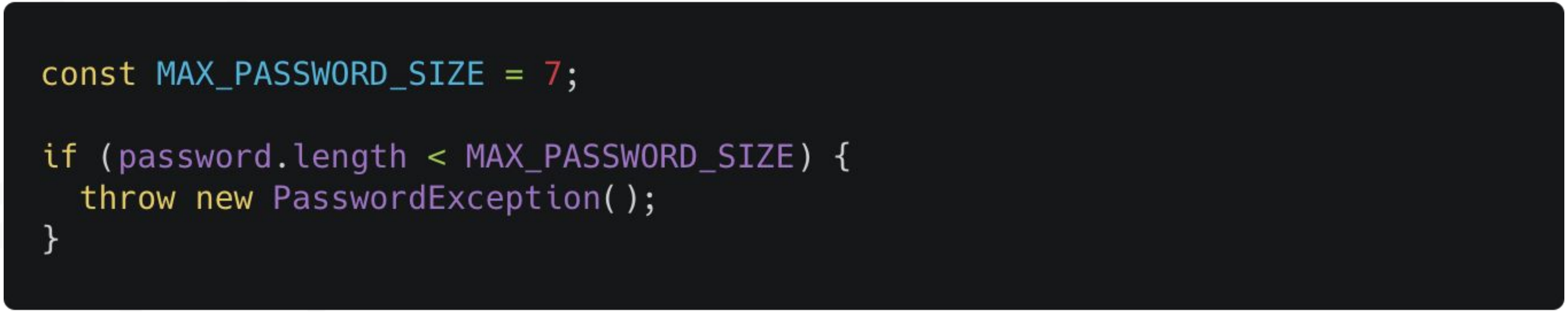
```
if (user.isActive()) {
            user.sendEmail();
    }
    if (user.isAdmin())
    {
        user.sendSlackNotification();
        }
else {
  logger.error('MY EYES');
}
```

Code

Magic numbers

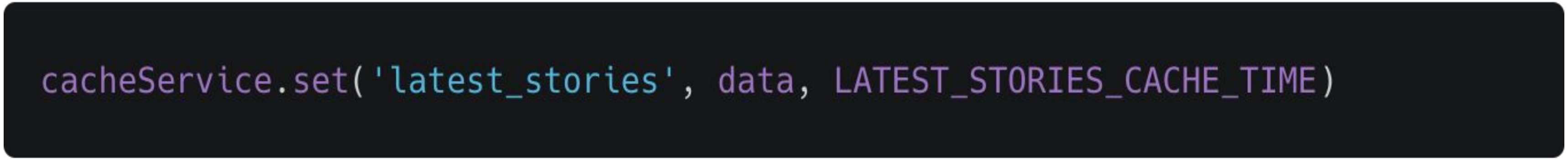


```
if (password.length < 7) {
  throw new PasswordException();
}
```

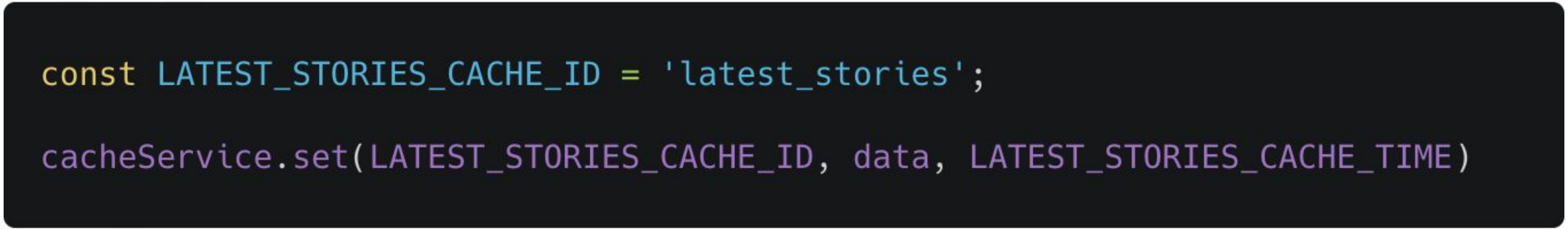


```
const MAX_PASSWORD_SIZE = 7;

if (password.length < MAX_PASSWORD_SIZE) {
  throw new PasswordException();
}
```

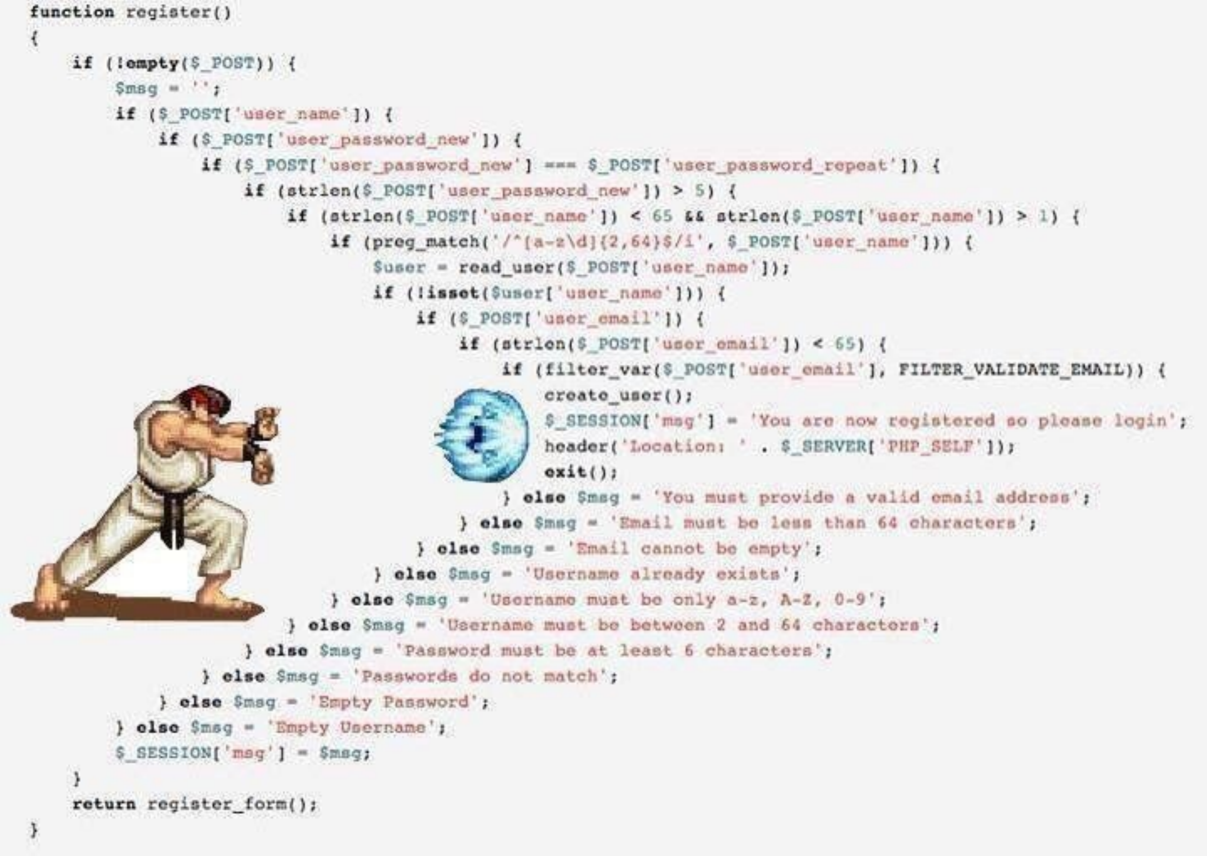
Magic strings



```
cacheService.set('latest_stories', data, LATEST_STORIES_CACHE_TIME)
```

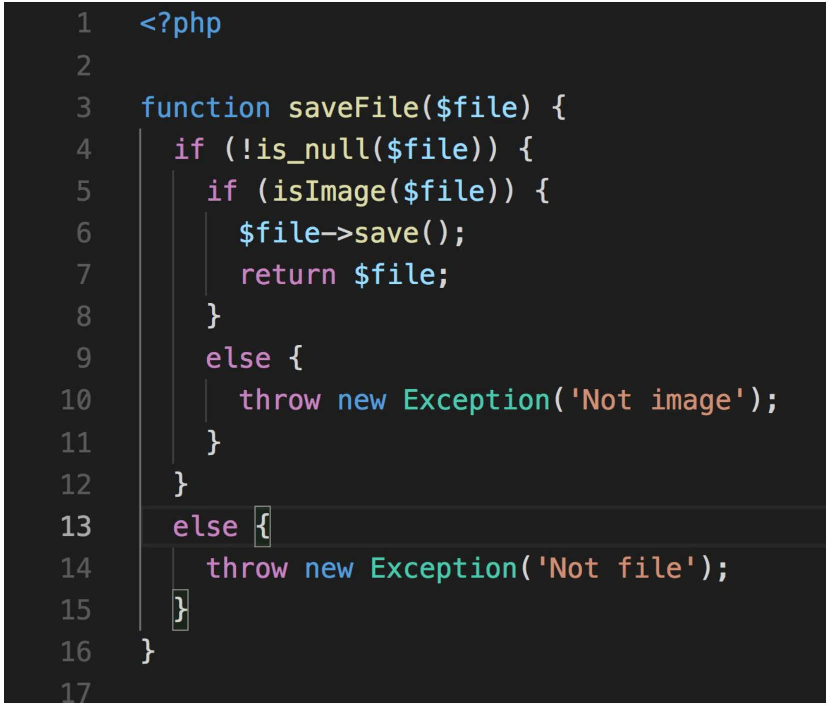


```
const LATEST_STORIES_CACHE_ID = 'latest_stories';

cacheService.set(LATEST_STORIES_CACHE_ID, data, LATEST_STORIES_CACHE_TIME)
```



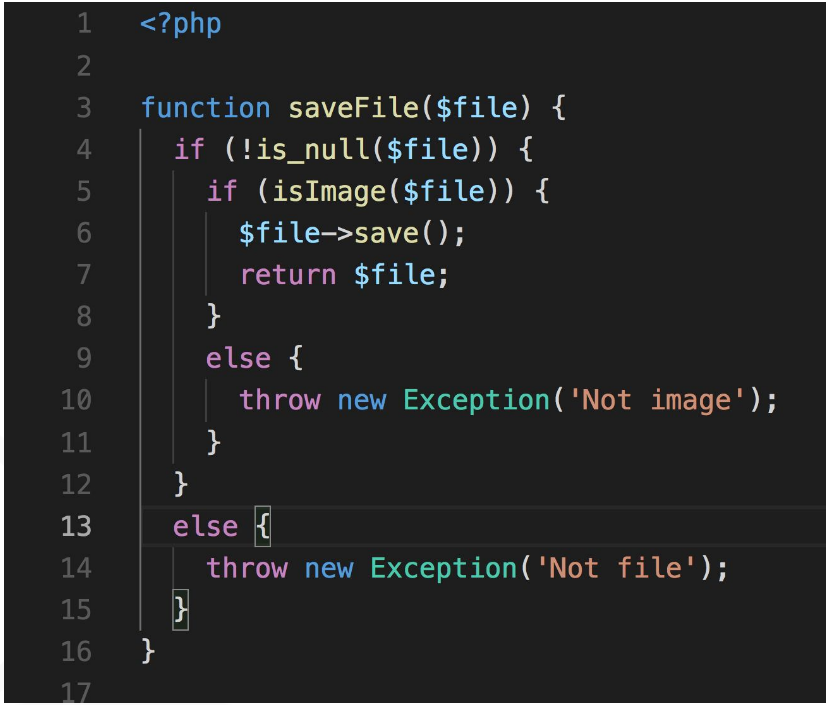
```
function register()
{
    if (!empty($_POST)) {
        $msg = '';
        if ($_POST['user_name']) {
            if ($_POST['user_password_new']) {
                if ($_POST['user_password_new'] === $_POST['user_password_repeat']) {
                    if (strlen($_POST['user_password_new']) > 5) {
                        if (strlen($_POST['user_name']) < 65 && strlen($_POST['user_name']) > 1) {
                            if (preg_match('/^[a-z\d]{2,64}$/i', $_POST['user_name'])) {
                                $user = read_user($_POST['user_name']);
                                if (!isset($user['user_name'])) {
                                    if ($_POST['user_email']) {
                                        if (strlen($_POST['user_email']) < 65) {
                                            if (filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
                                                create_user();
                                                $_SESSION['msg'] = 'You are now registered so please login';
                                                header('Location: ' . $_SERVER['PHP_SELF']);
                                                exit();
                                            } else $msg = 'You must provide a valid email address';
                                        } else $msg = 'Email must be less than 64 characters';
                                    } else $msg = 'Email cannot be empty';
                                } else $msg = 'Username already exists';
                            } else $msg = 'Username must be only a-z, A-Z, 0-9';
                        } else $msg = 'Username must be between 2 and 64 characters';
                    } else $msg = 'Password must be at least 6 characters';
                } else $msg = 'Passwords do not match';
            } else $msg = 'Empty Password';
        } else $msg = 'Empty Username';
        $_SESSION['msg'] = $msg;
    }
    return register_form();
}
```

If - else

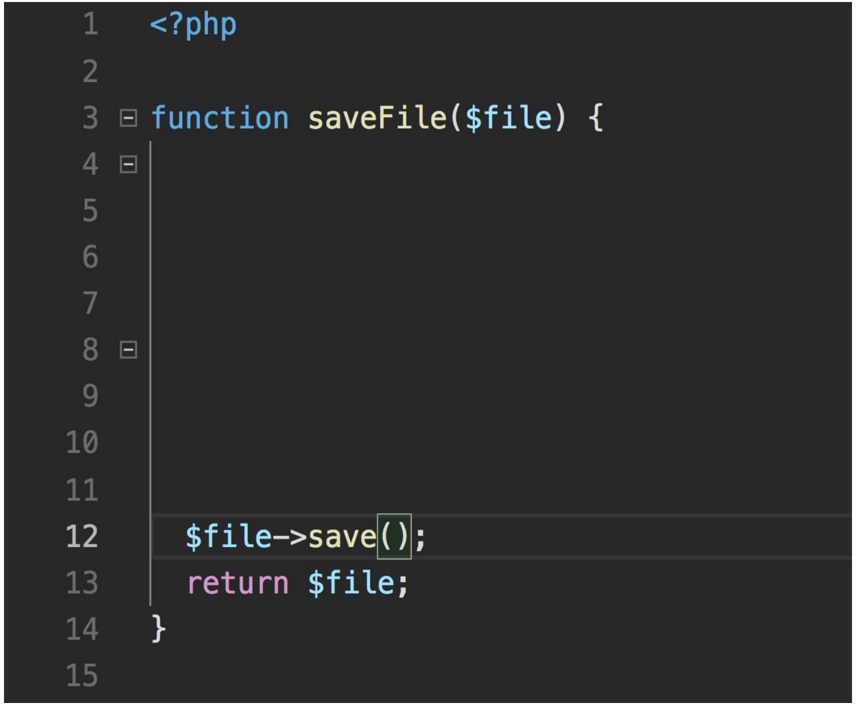


```php
<?php

function saveFile($file) {
  if (!is_null($file)) {
    if (isImage($file)) {
      $file->save();
      return $file;
    }
    else {
      throw new Exception('Not image');
    }
  }
  else {
    throw new Exception('Not file');
  }
}
```

If - else

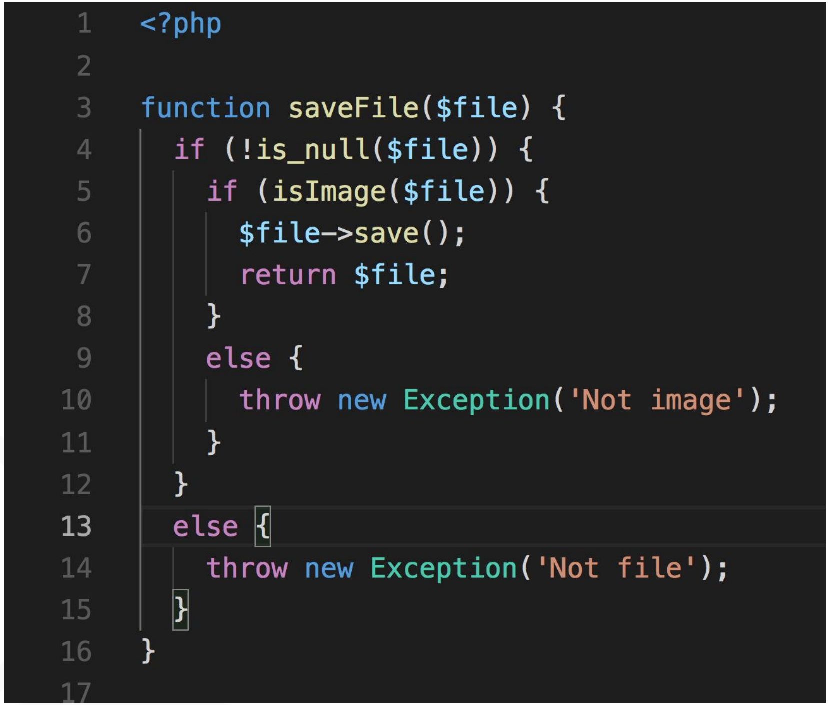


```php
<?php

function saveFile($file) {
  if (!is_null($file)) {
    if (isImage($file)) {
      $file->save();
      return $file;
    }
    else {
      throw new Exception('Not image');
    }
  }
  else {
    throw new Exception('Not file');
  }
}
```

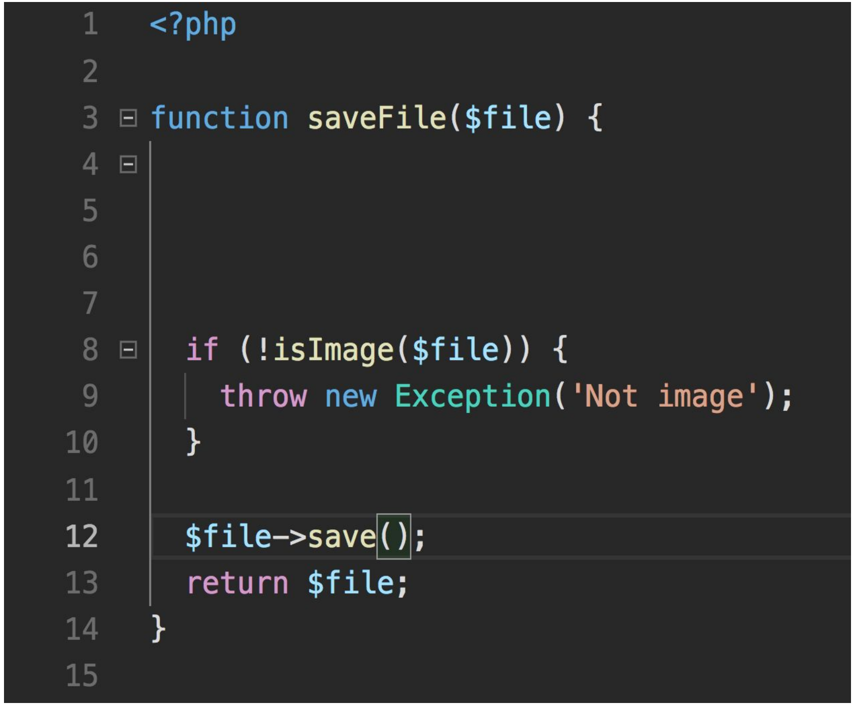


```php
<?php

function saveFile($file) {








  $file->save();
  return $file;
}

```

If - else

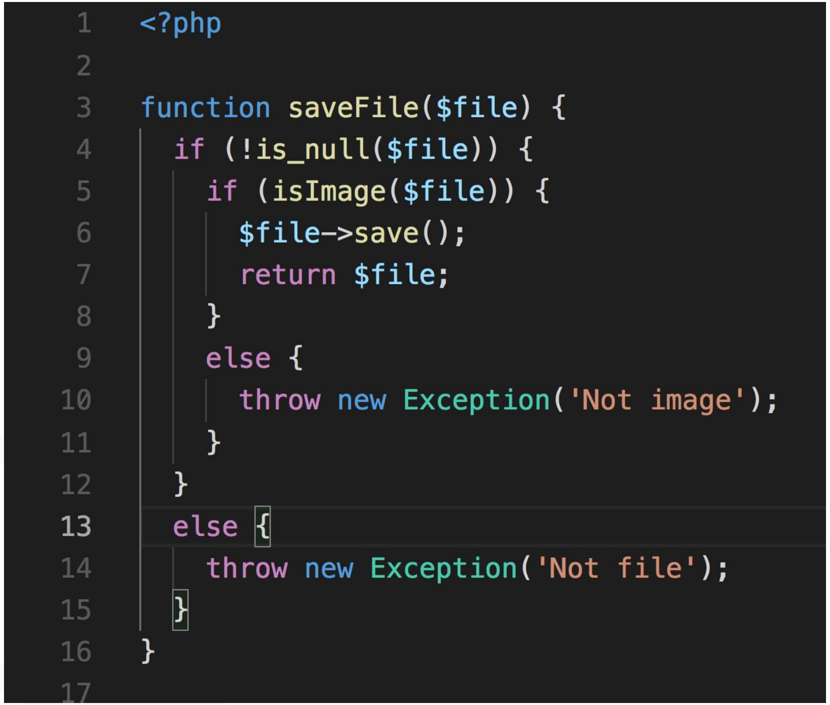


```php
<?php

function saveFile($file) {
  if (!is_null($file)) {
    if (isImage($file)) {
      $file->save();
      return $file;
    }
    else {
      throw new Exception('Not image');
    }
  }
  else {
    throw new Exception('Not file');
  }
}
```

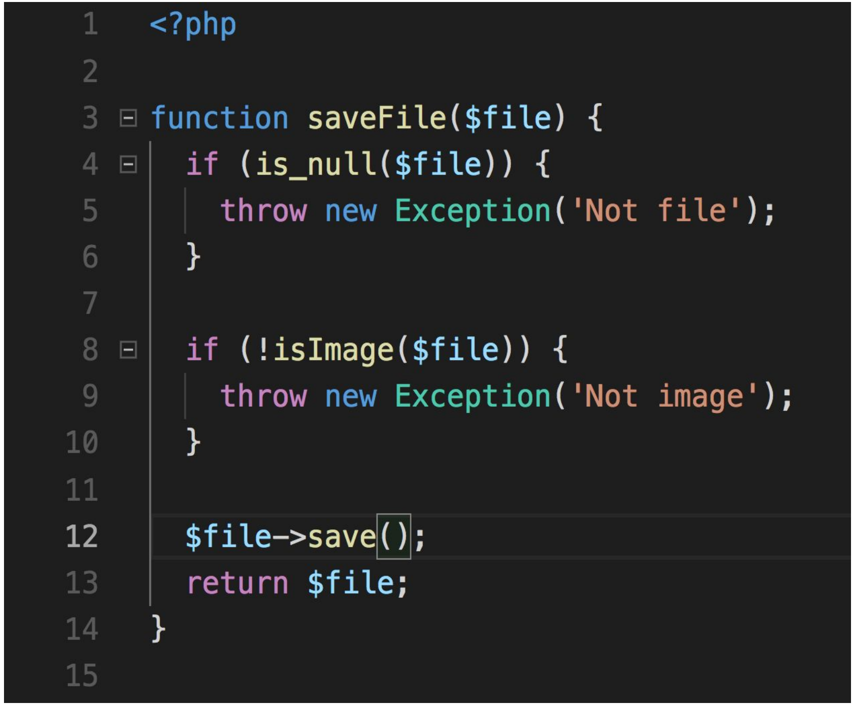


```php
<?php

function saveFile($file) {




  if (!isImage($file)) {
    throw new Exception('Not image');
  }

  $file->save();
  return $file;
}
```

If - else

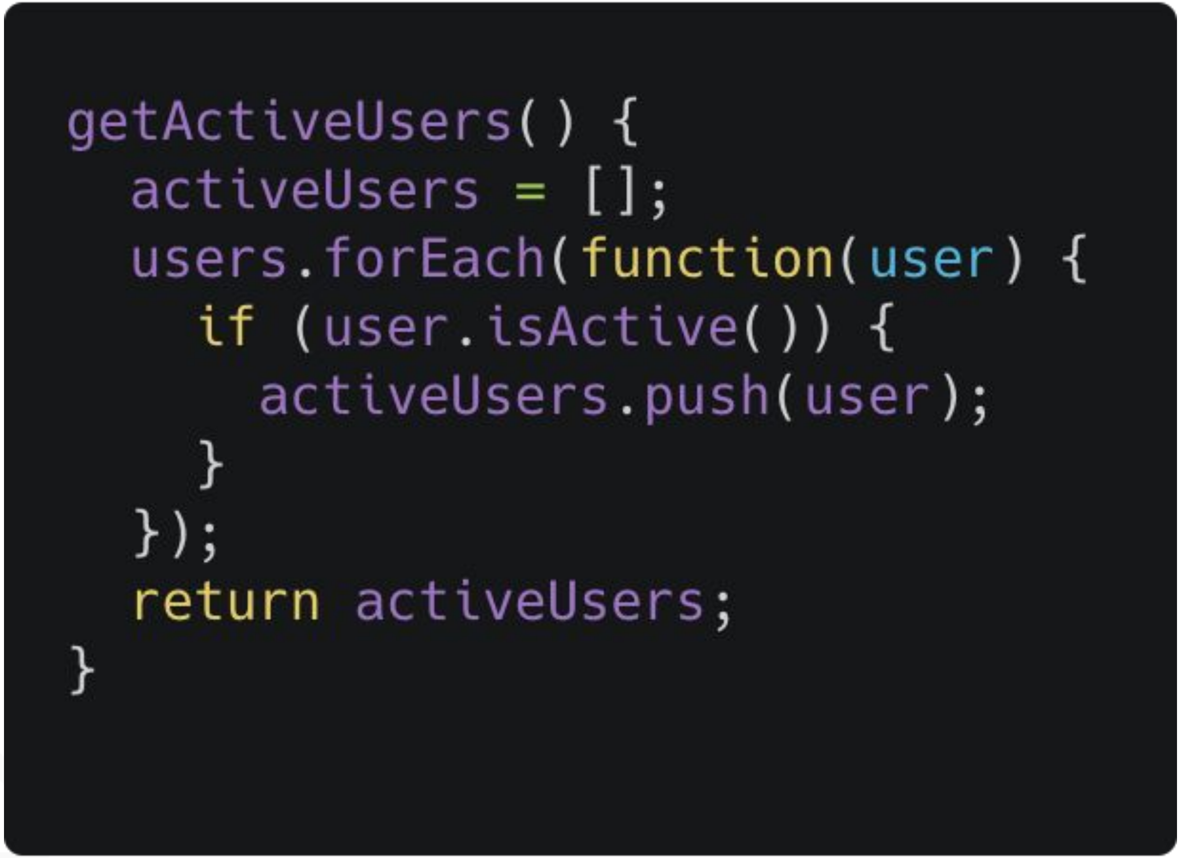


```php
<?php

function saveFile($file) {
  if (!is_null($file)) {
    if (isImage($file)) {
      $file->save();
      return $file;
    }
    else {
      throw new Exception('Not image');
    }
  }
  else {
    throw new Exception('Not file');
  }
}
```



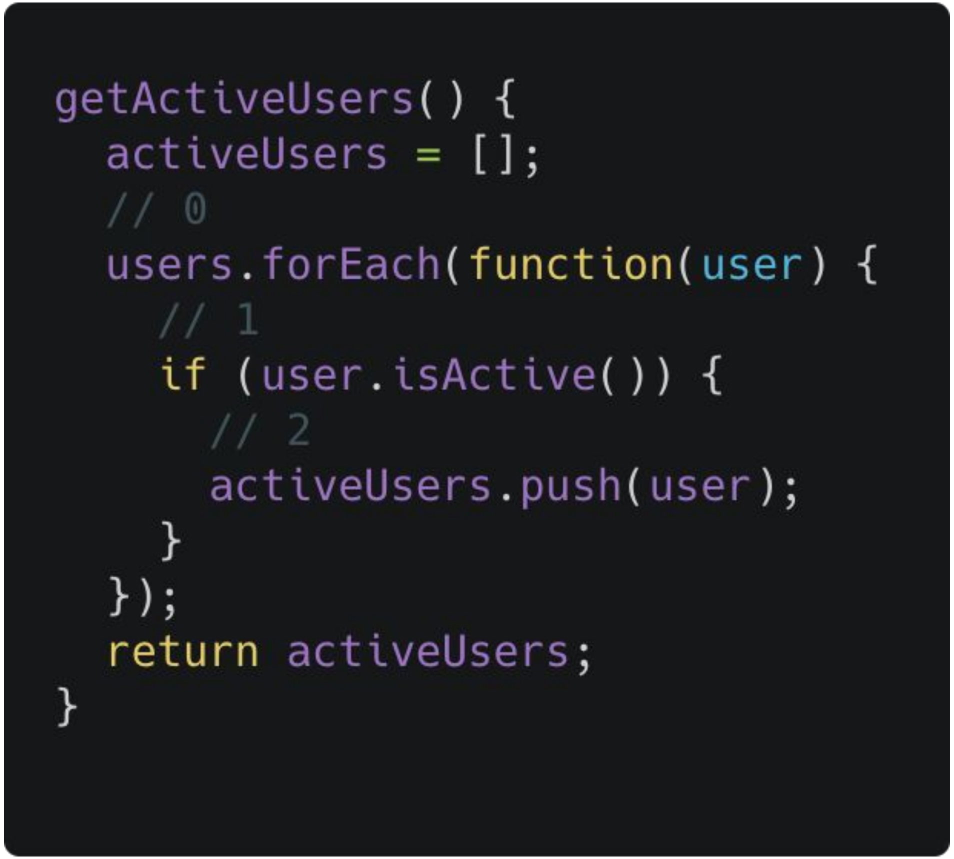
```php
<?php

function saveFile($file) {
  if (is_null($file)) {
    throw new Exception('Not file');
  }

  if (!isImage($file)) {
    throw new Exception('Not image');
  }

  $file->save();
  return $file;
}
```

One level of indentation



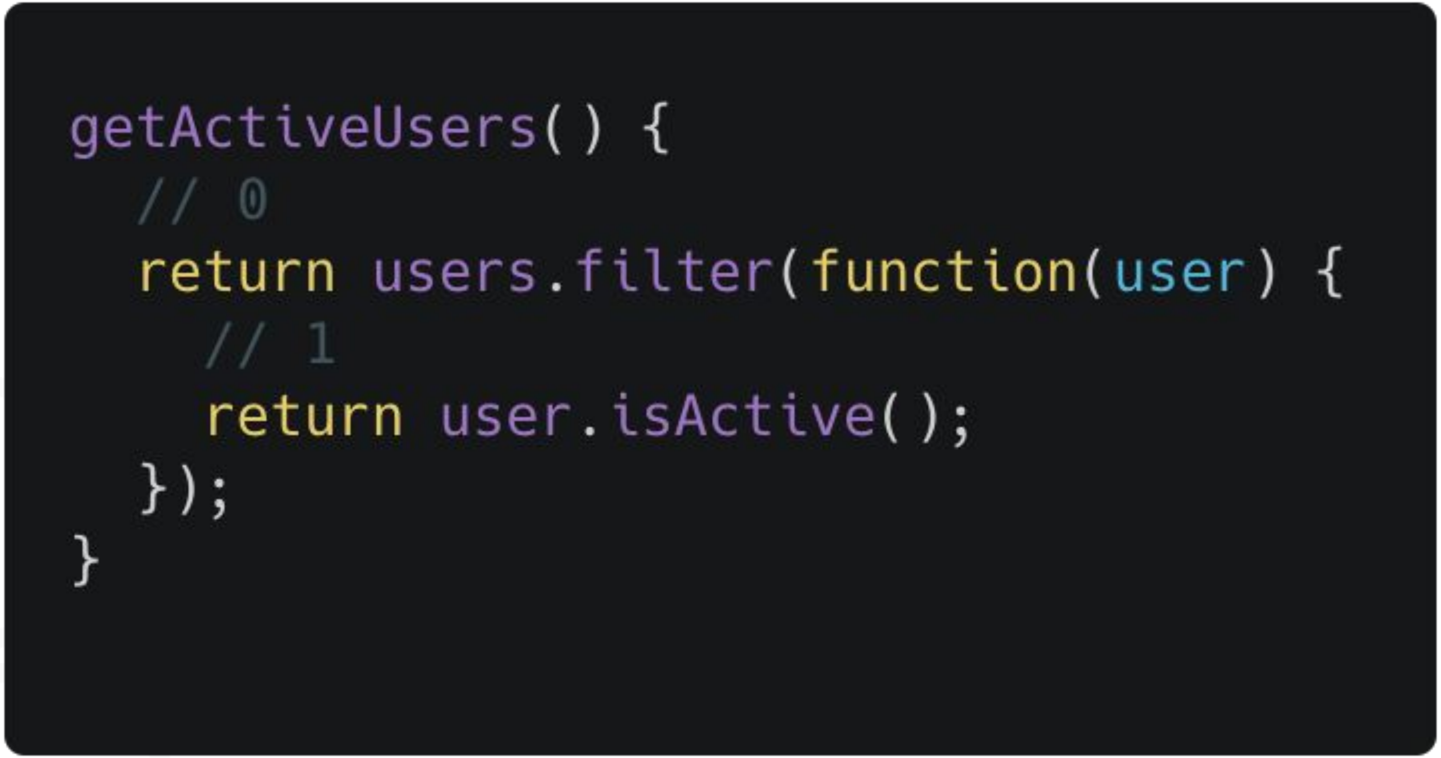
```
getActiveUsers() {
  activeUsers = [];
  users.forEach(function(user) {
    if (user.isActive()) {
      activeUsers.push(user);
    }
  });
  return activeUsers;
}
```

One level of indentation



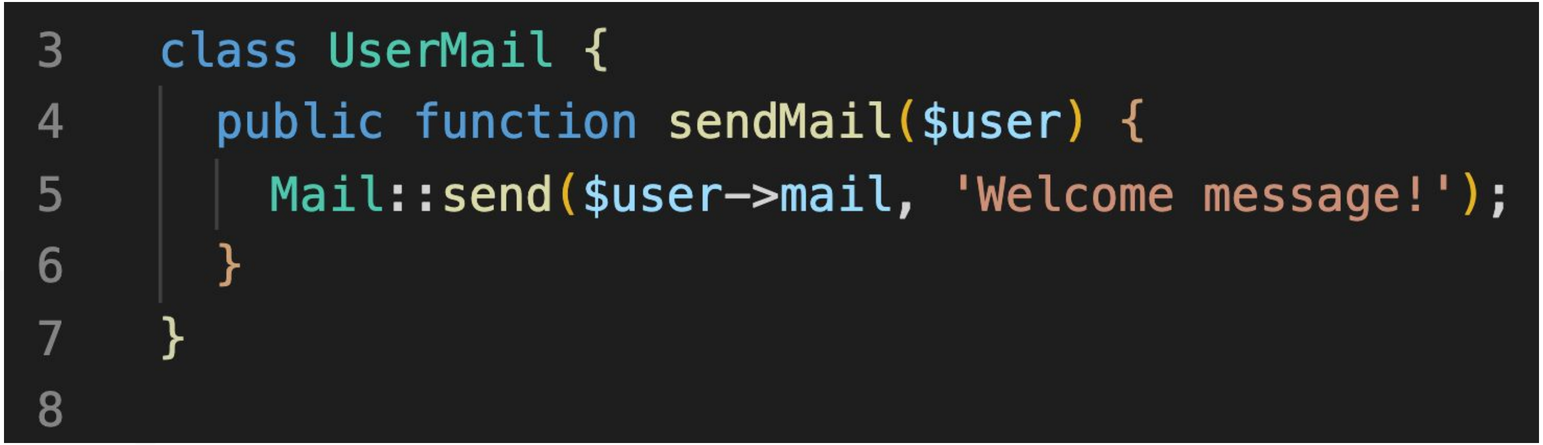
```
getActiveUsers() {
  activeUsers = [];
  // 0
  users.forEach(function(user) {
    // 1
    if (user.isActive()) {
      // 2
      activeUsers.push(user);
    }
  });
  return activeUsers;
}
```

One level of indentation



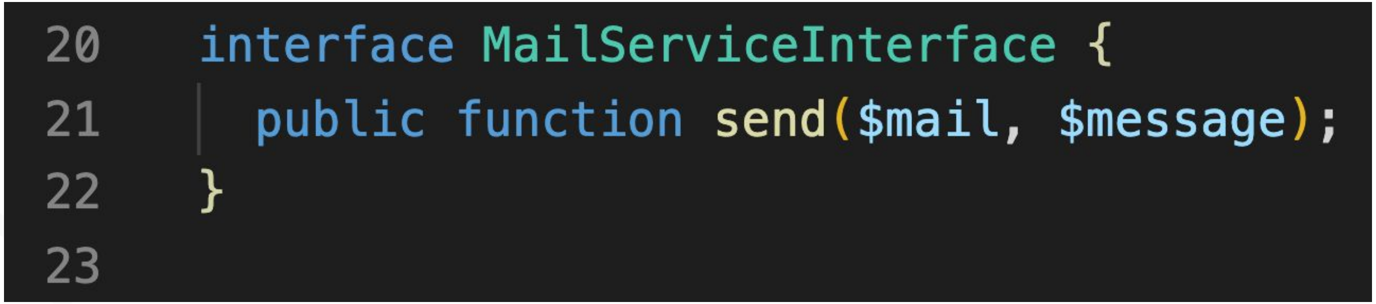
```
getActiveUsers() {
  // 0
  return users.filter(function(user) {
    // 1
    return user.isActive();
  });
}
```

Dependency injection



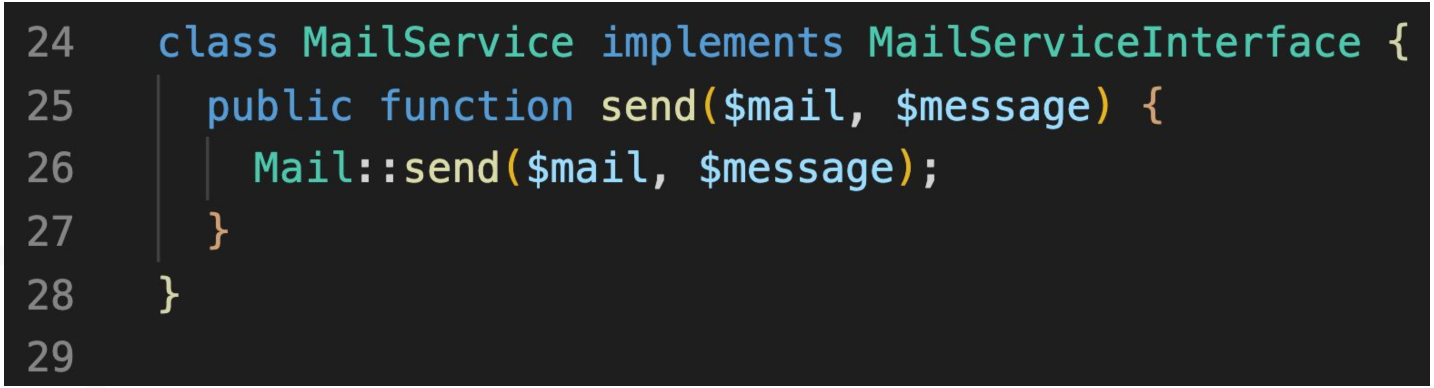
```
class UserMail {
  public function sendMail($user) {
    Mail::send($user->mail, 'Welcome message!');
  }
}

```

Dependency injection



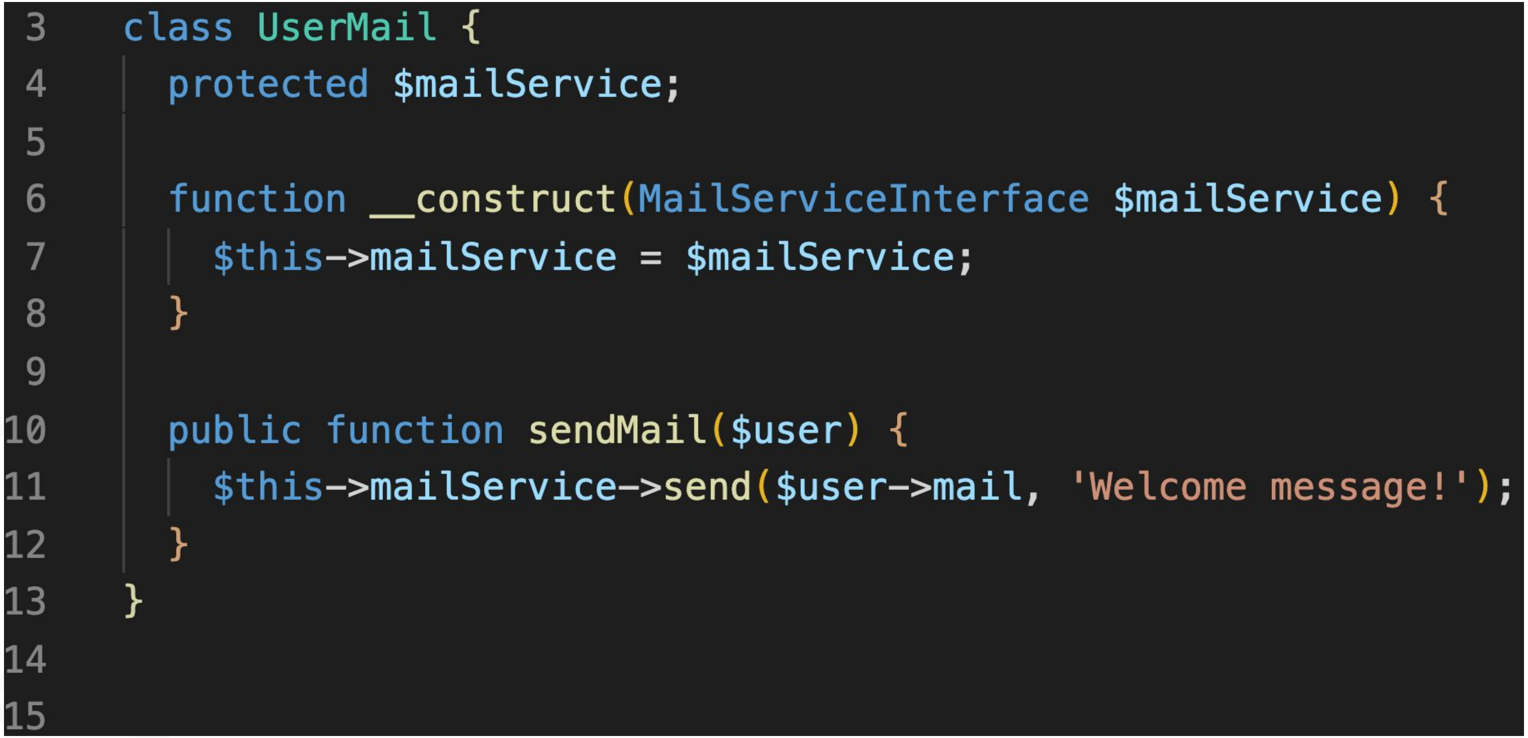
```
interface MailServiceInterface {
  public function send($mail, $message);
}

```

Dependency injection



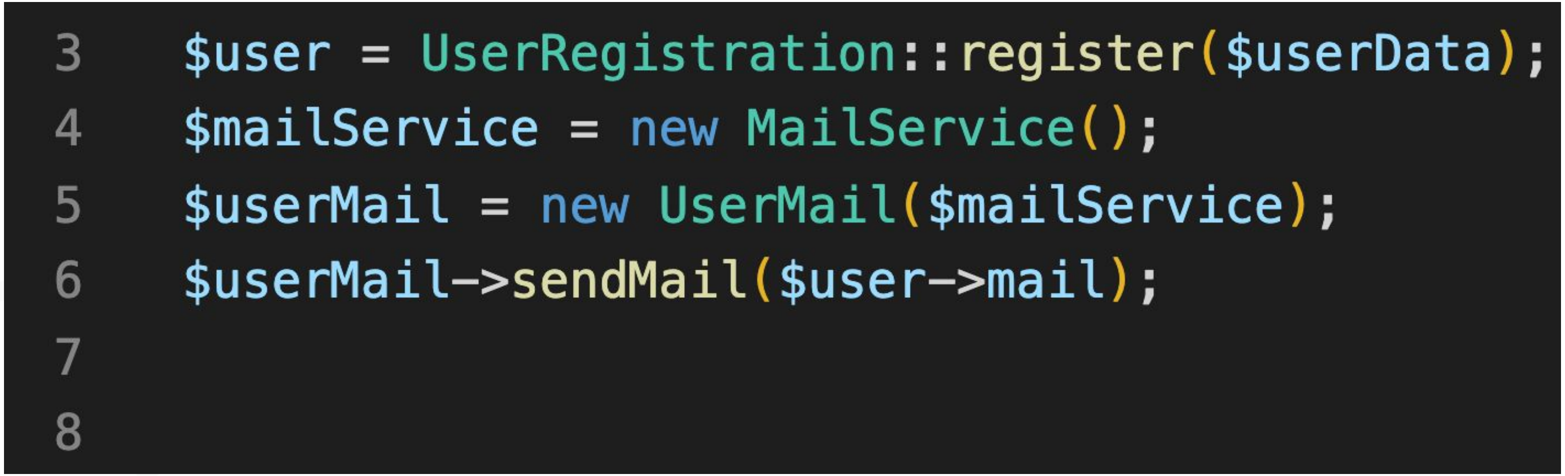
```php
class MailService implements MailServiceInterface {
  public function send($mail, $message) {
    Mail::send($mail, $message);
  }
}

```

Dependency injection



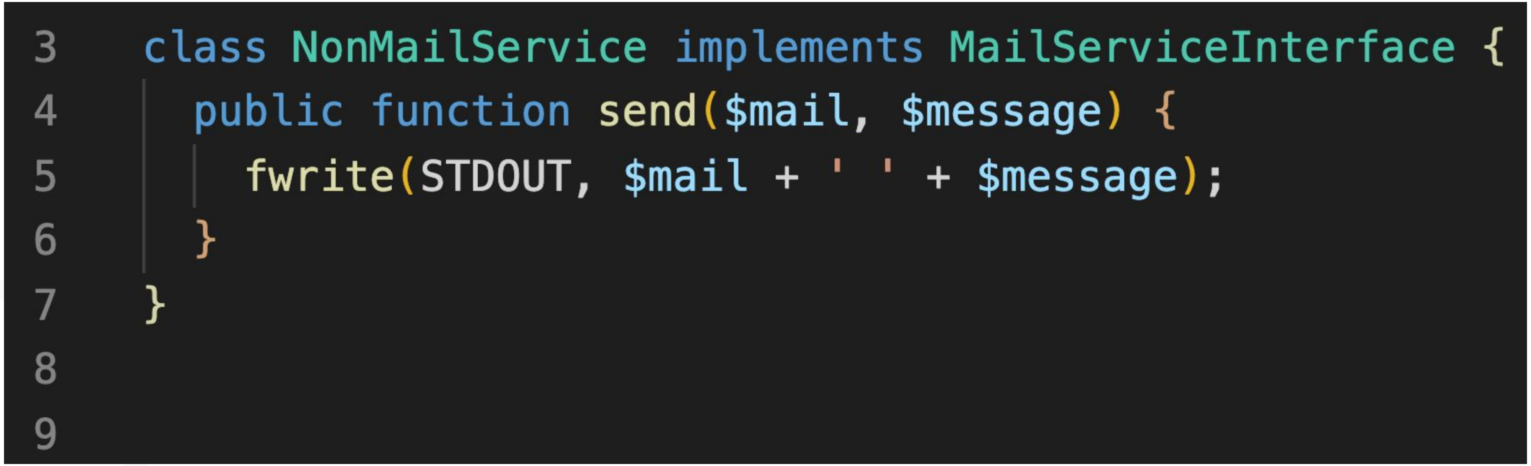
```php
class UserMail {
  protected $mailService;

  function __construct(MailServiceInterface $mailService) {
    $this->mailService = $mailService;
  }

  public function sendMail($user) {
    $this->mailService->send($user->mail, 'Welcome message!');
  }
}


```

Dependency injection



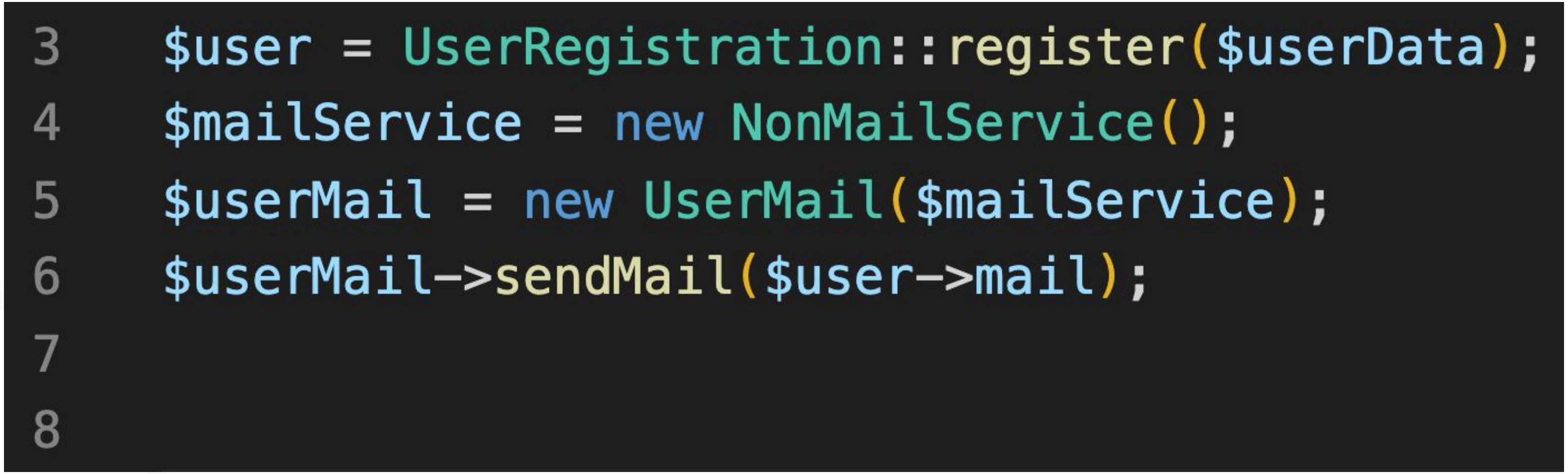
```php
$user = UserRegistration::register($userData);
$mailService = new MailService();
$userMail = new UserMail($mailService);
$userMail->sendMail($user->mail);


```

Dependency injection



```php
class NonMailService implements MailServiceInterface {
  public function send($mail, $message) {
    fwrite(STDOUT, $mail + ' ' + $message);
  }
}


```

Dependency injection



```
$user = UserRegistration::register($userData);
$mailService = new NonMailService();
$userMail = new UserMail($mailService);
$userMail->sendMail($user->mail);


```

Practice

THANKS
I GUESS
memegenerator.net

Ask questions!

?