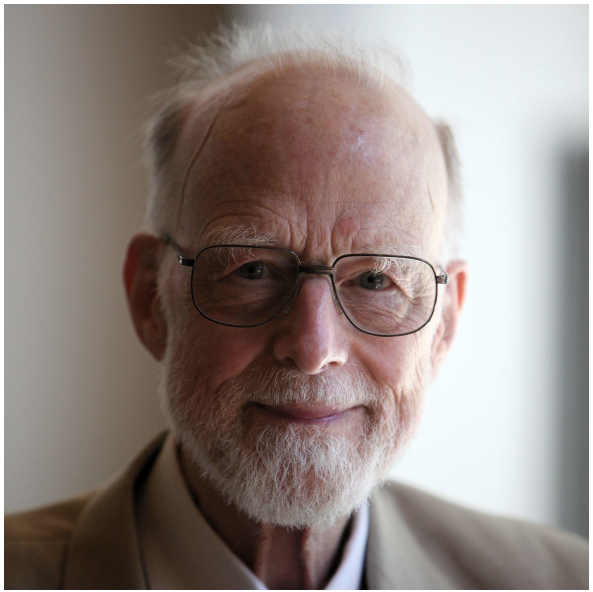
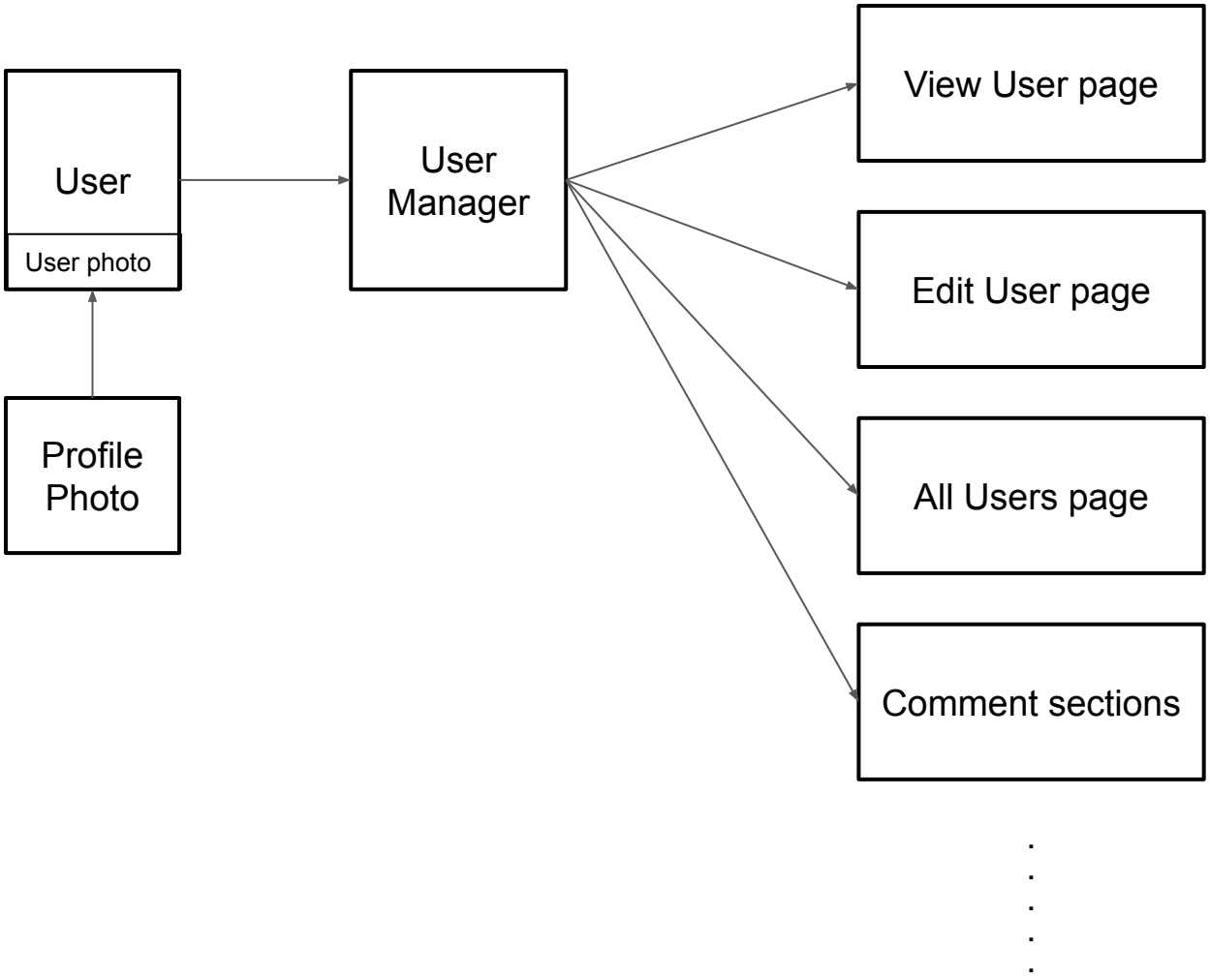
NOT NULL



Tony Hoare

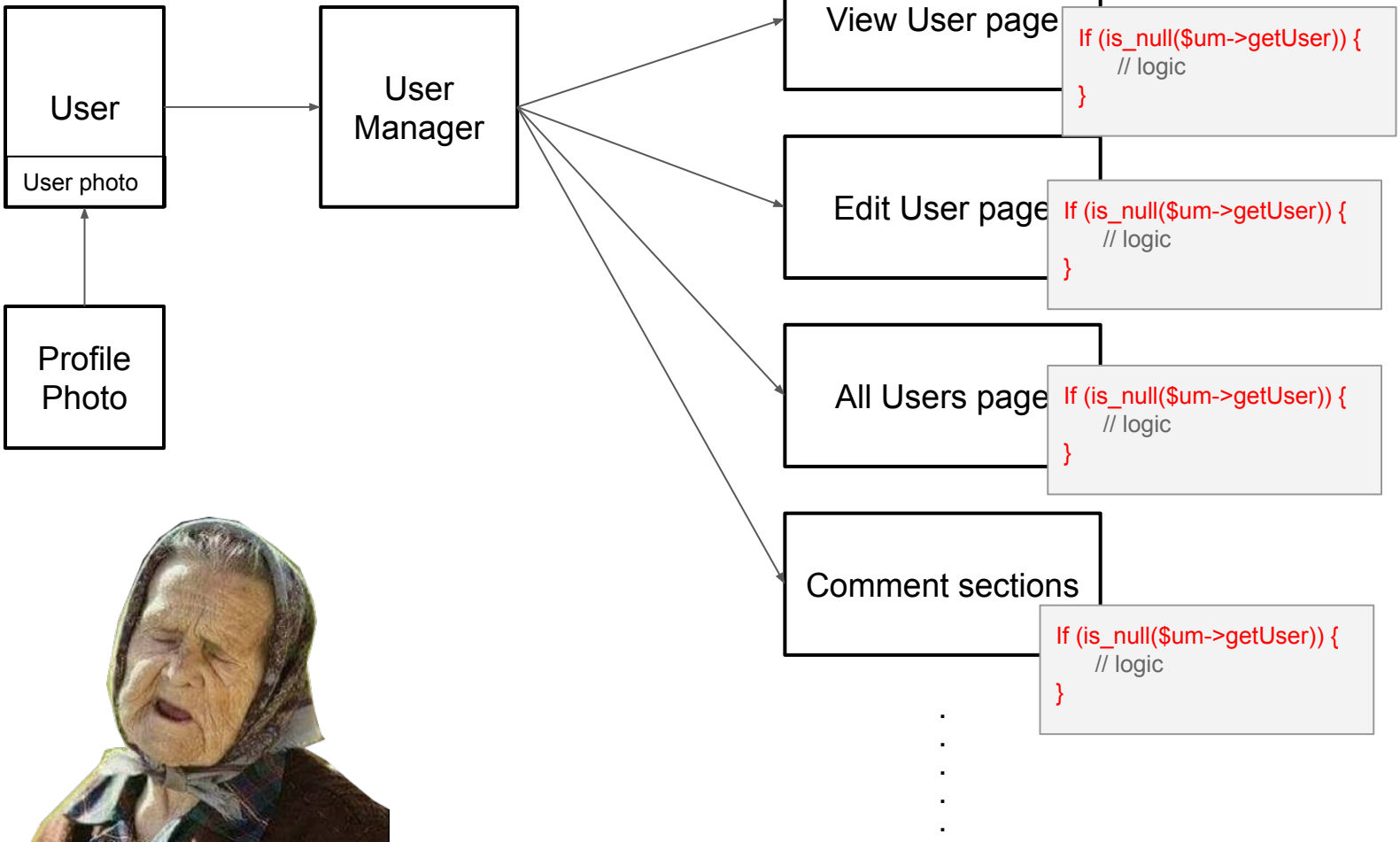
In 2009, he apologised for inventing the null reference.

“I call it my billion-dollar mistake. “


User
User photo
Profile Photo
User Manager
View User page
Edit User page
All Users page
Comment sections
.
.
.
.
.

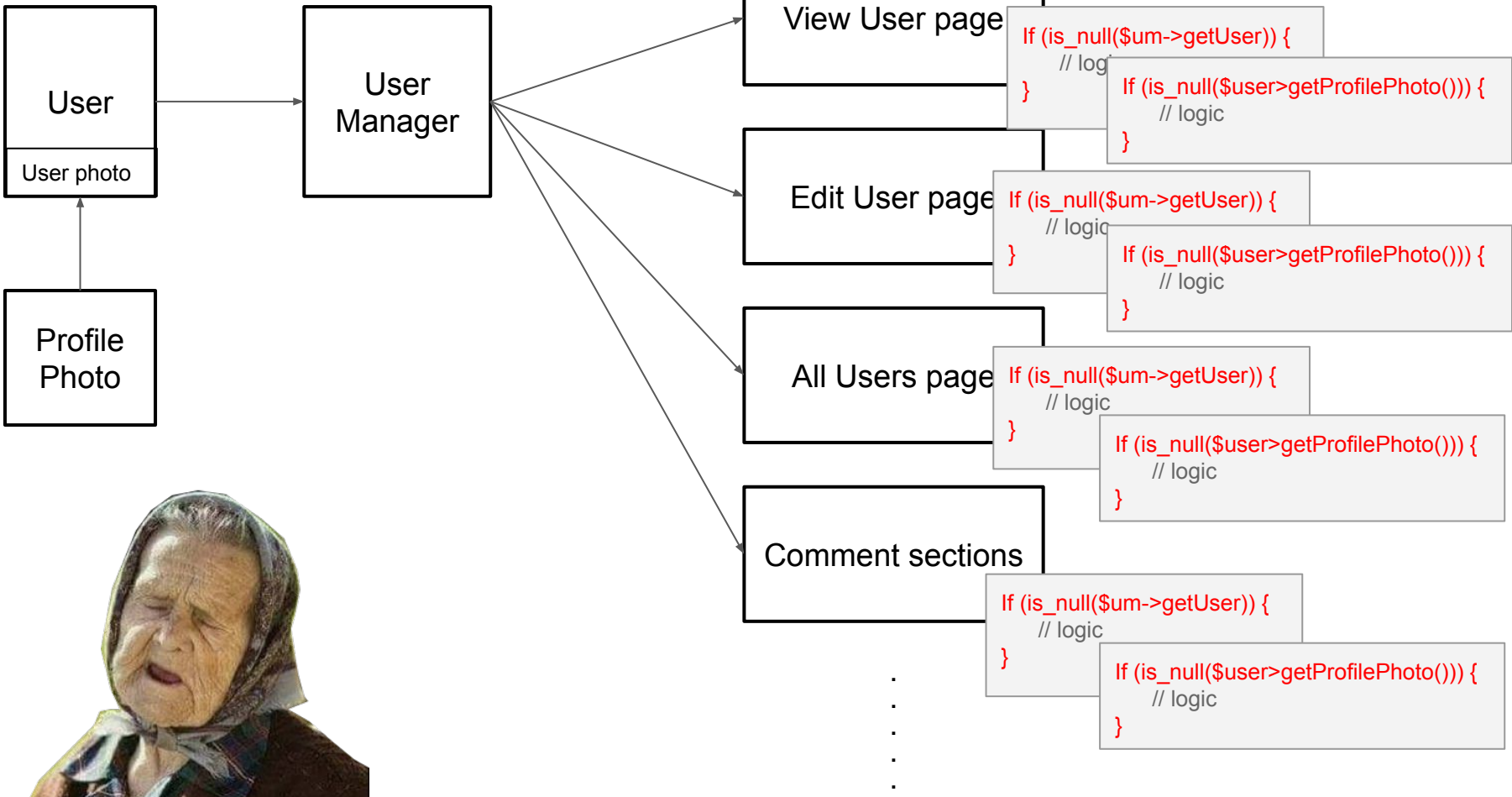
```php
class User
{
	private $profilePhoto;

	private $userName;

	public function getUserName(): ?string
	{
		return $this->userName;
	 }

	public function getProfilePhoto(): ?ProfilePhoto
	{
		return $this->profilePhoto;
	}

	public function setProfilePhoto(?ProfilePhoto $profilePhoto)
	{
		$this->profilePhoto = $profilePhoto;
	}
}


class UserManager
{

	public function find(): ?User
	{
		/** IMAGINE CODE HERE*/
	}
}
```

```php
class ProfilePhoto
{
	private $url;

	private $width;

	private $height;

	public function getUrl(): ?string
	{
		return $this->url;
	}

	public function getWidth(): ?int
	{
		return $this->width;
	}

	public function getHeight(): ?int
	{
		return $this->height;
	}

	/**
	* IMAGINE SETTERS HERE
	*/
}
```


User
User photo
Profile Photo
User Manager
View User page
If (is_null($um->getUser)) {
// logic
}
Edit User page
If (is_null($um->getUser)) {
// logic
}
All Users page
If (is_null($um->getUser)) {
// logic
}
Comment sections
If (is_null($um->getUser)) {
// logic
}

```php
class Example
{
	private const DEFAULT_USERNAME = 'Damnjan';

	private const DEFAULT_PROFILE_PHOTO = '/var/img/default.jpg';

	public function getUser($userId): ?User
	{
		return $this->getRepository()->find($userId);
	}

	public function getUserData($userId)
	{
		$user = $this->getUser($userId);

		$photoUrl = self::DEFAULT_PROFILE_PHOTO;
		$userName = self::DEFAULT_USERNAME;

		if (null !== $user) {
			$profilePhoto = $user->getProfilePhoto();

			if (null !== $profilePhoto) {
				$photoUrl = $profilePhoto->getUrl();
			}

			if (!is_null($user->getUserName())) {
				$userName =  $user->getUserName();
			}
		}
		/* ..... */
	}
}
```

```php
class TestExample
{
	public function testGetUserDataIfUserDoesNotExist()
	{
		/** ..... */
	}

	public function testGetUserDataIfProfilePhotoMissing()
	{
		/** ..... */
	}

	public function testGetUserDataIfProfilePhotoUrlMissing()
	{
		/** ..... */
	}

	public function testGetUserDataIfUserNameMissing()
	{
		/** ..... */
	}

	public function testGetUserData()
	{
		/** ..... */
	}
}
```

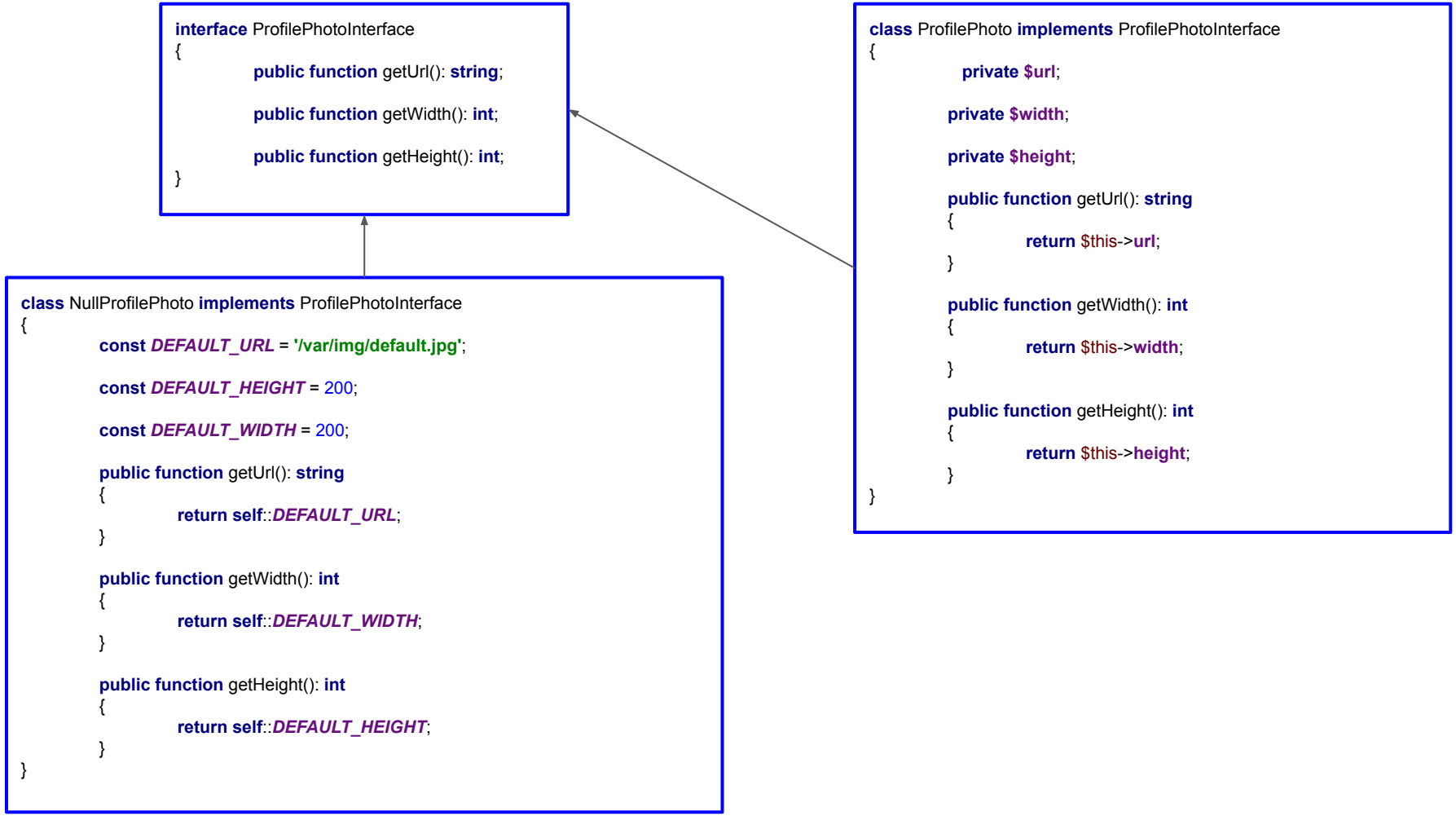

User
User photo
Profile Photo
User Manager
View User page
If (is_null($um->getUser)) {
// log
}
If (is_null($user>getProfilePhoto())) {
// logic
}
Edit User page
If (is_null($um->getUser)) {
// logic
}
If (is_null($user>getProfilePhoto())) {
// logic
}
All Users page
If (is_null($um->getUser)) {
// logic
}
If (is_null($user>getProfilePhoto())) {
// logic
}
Comment sections
If (is_null($um->getUser)) {
// logic
}
If (is_null($user>getProfilePhoto())) {
// logic
}

DEFENSIVE PROGRAMMING

DEFECTS

- No encapsulations
- Untrusted objects
- Check has to be performed every time
- Tell don’t ask can’t be applied

NULL OBJECT PATTERN


interface ProfilePhotoInterface
{
public function getUrl(): string;
public function getWidth(): int;
public function getHeight(): int;
}
class ProfilePhoto implements ProfilePhotoInterface
{
private $url;
private $width;
private $height;
public function getUrl(): string
{
return $this->url;
}
public function getWidth(): int
{
return $this->width;
}
public function getHeight(): int
{
return $this->height;
}
}
class NullProfilePhoto implements ProfilePhotoInterface
{
const DEFAULT_URL = '/var/img/default.jpg';
const DEFAULT_HEIGHT = 200;
const DEFAULT_WIDTH = 200;
public function getUrl(): string
{
return self::DEFAULT_URL;
}
public function getWidth(): int
{
return self::DEFAULT_WIDTH;
}
public function getHeight(): int
{
return self::DEFAULT_HEIGHT;
}
}

```php
class User {

	private $profilePhoto;

	private $userName;

	public function getUserName(): ?string
	{
		return $this->userName;
	}

	public function getProfilePhoto(): ProfilePhotoInterface
	{
		if (null === $this->profilePhoto) {
			return new NullProfilePhoto();
		}

		return $this->profilePhoto;
	}

	public function setProfilePhoto(ProfilePhotoInterface $profilePhoto)
	{
		$this->profilePhoto = $profilePhoto;
	}
}
```

User

```
If (is_null($user>getProfilePhoto())) {
    // logic
}
```

Profile Photo

User Manager

View User page

```
If (is_null($um->getUser)) {
    // logic
}
```

Edit User page

```
If (is_null($um->getUser)) {
    // logic
}
```

All Users page

```
If (is_null($um->getUser)) {
    // logic
}
```

Comment sections

```
If (is_null($um->getUser)) {
    // logic
}
```

. . . . .

SETTING DEFAULTS

```php
class ProfilePhoto implements ProfilePhotoInterface
{
	private $url;

	private $width;

	private $height;

	public function getUrl(): string
	{
		 if (null === $this->url) {
			return NullProfilePhoto::DEFAULT_URL;
		}

		return $this->url;
	}

	public function getWidth(): int
	{
		if (null === $this->width) {
			return NullProfilePhoto::DEFAULT_WIDTH;
		}

		return $this->width;
	}

	public function getHeight(): int
	{
		if (null === $this->height) {
			return NullProfilePhoto::DEFAULT_HEIGHT;
		}

		return $this->height;
	}
}
```

```php
class User {

	const DEFAULT_USERNAME = 'Damnjan';

	private $profilePhoto;

	private $userName;

	public function getUserName(): string
	{
		if (null === $this->userName) {
			return self::DEFAULT_USERNAME;
		}

		return $this->userName;
	}

	public function getProfilePhoto(): ProfilePhotoInterface
	{
		if (null === $this->profilePhoto) {
			return new NullProfilePhoto();
		}

		return $this->profilePhoto;
	}

	public function setProfilePhoto(ProfilePhotoInterface $profilePhoto)
	{
		$this->profilePhoto = $profilePhoto;
	}
}
```

```php
class Example
{
	public function getUser($userId): ?User
	{
		return $this->getRepository()->find($userId);
	}

	public function getUserData($userId)
	{
		$user = $this->getUser($userId);

		$photoUrl = NullProfilePhoto::DEFAULT_URL;
		$userName = User::DEFAULT_USERNAME;

		if (null !== $user) {
			$photoUrl = $user->getProfilePhoto()->getUrl();
			$userName =  $user->getUserName();
		}

		/* ..... */
	}
}
```

```php
class TestExample
{
	public function testGetUserDataIfUserDoesNotExist()
	{
		/** ..... */
	}

	public function testGetUserData()
	{
		/** ..... */
	}
}
```

THROWING EXCEPTION

```php
class Example
{
	/**
	* @throws \Exception
	*/
	public function getUser($userId): User
	{
		return $this->getRepository()->find($userId);
	}

	public function getUserData($userId)
	{
		try {
			$user = $this->getUser($userId);

			$photoUrl = $user->getProfilePhoto()->getUrl();
			$userName =  $user->getUserName();
			/* ..... */
		} catch (\Exception $exception) {
			return $this->getDefaultUserData();
		}
	}

	private function getDefaultUserData()
	{
		$photoUrl = NullProfilePhoto::DEFAULT_URL;
		$userName = User::DEFAULT_USERNAME;
		/* ..... */
	}
}
```

```php
class TestExample
{
	public function testGetUserDataIfUserDoesNotExist()
	{
		/** ..... */
	}

	public function testGetUserData()
	{
		/** ..... */
	}
}
```

User

```
If (is_null($user>getProfilePhoto())) {
      // logic
}
```

Profile Photo

User Manager

```
If (is_null($um->getUser)) {
      // logic
}
```

View User page

Try catch

Edit User page

Try catch

All Users page

Try catch

Comment sections

Try catch

.
.
.
.
.

CONTAINER OBJECT
OPTIONAL OBJECT

```php
class ProfilePhotoContainer
{
	private $profilePhoto;

	public function __construct(?ProfilePhoto $profilePhoto)
	{
		$this->profilePhoto = $profilePhoto;
	}

	public function doExist()
	{
		return (null !== $this->profilePhoto);
	}

	public function getProfilePhoto(): ?ProfilePhoto
	{
		return $this->profilePhoto;
	}
}
```

```php
class User {

	private $profilePhoto;

	private $userName;

	public function getUserName(): string
	{
		return $this->userName;
	}

	public function getProfilePhoto(): ProfilePhotoContainer
	{
		return new ProfilePhotoContainer($this->profilePhoto);
	}

	public function setProfilePhoto(ProfilePhoto $profilePhoto)
	{
		$this->profilePhoto = $profilePhoto;
	}
}
```

WHAT IF WE REALLY NEED TO KNOW REAL VALUE?

WHAT IF WE REALLY NEED TO CHECK IS IT NULL?

```php
class ProfilePhoto implements ProfilePhotoInterface
{
	private $url;

	private $width;

	public function isUrlNull(): bool
	{
		return (null === $this->url);
	}

	public function getRawWidth(): ?int
	{
		return $this->width;
	}

	public function getUrl(): string
	{
		if (null === $this->url) {
			return NullProfilePhoto::DEFAULT_URL;
		}

		return $this->url;
	}

	public function getWidth(): int
	{
		if (null === $this->width) {
			return NullProfilePhoto::DEFAULT_WIDTH;
		}

		return $this->width;
	}
}
```

THANK YOU

@damnjan

http://dev.to/damnjan