ELMO Cloud HR & Payroll

Living with the Legacy

Dave Newson | Software Engineer

I do these things

ELMO
CLOUD HR & PAYROLL

Work with awesome people

App development

Build libraries

Debug real-world issues

Under estimate tasks

Virtual Machines

Typescript

Microservice design

I do these things too

ELMO
CLOUD HR & PAYROLL

Game design

Work with awesome people

App development

Build libraries

NodeJS

Debug real-world issues

Under estimate tasks

React

Reticulate Splines

Virtual Machines

Word clouds

Docker

C#

Typescript

Push back deadlines

Microservice design

Security testing

Mostly I do this

ELMO
CLOUD HR & PAYROLL

<?php

ELMO
CLOUD HR & PAYROLL

What is legacy?


“Everything the user touches is Legacy”

What is legacy?

ELMO
CLOUD HR & PAYROLL

- If it’s in use by a customer it is part of the legacy
- We’re constantly building on these foundations
- Everything has an ongoing maintenance cost
- Try to ship sustainable systems!

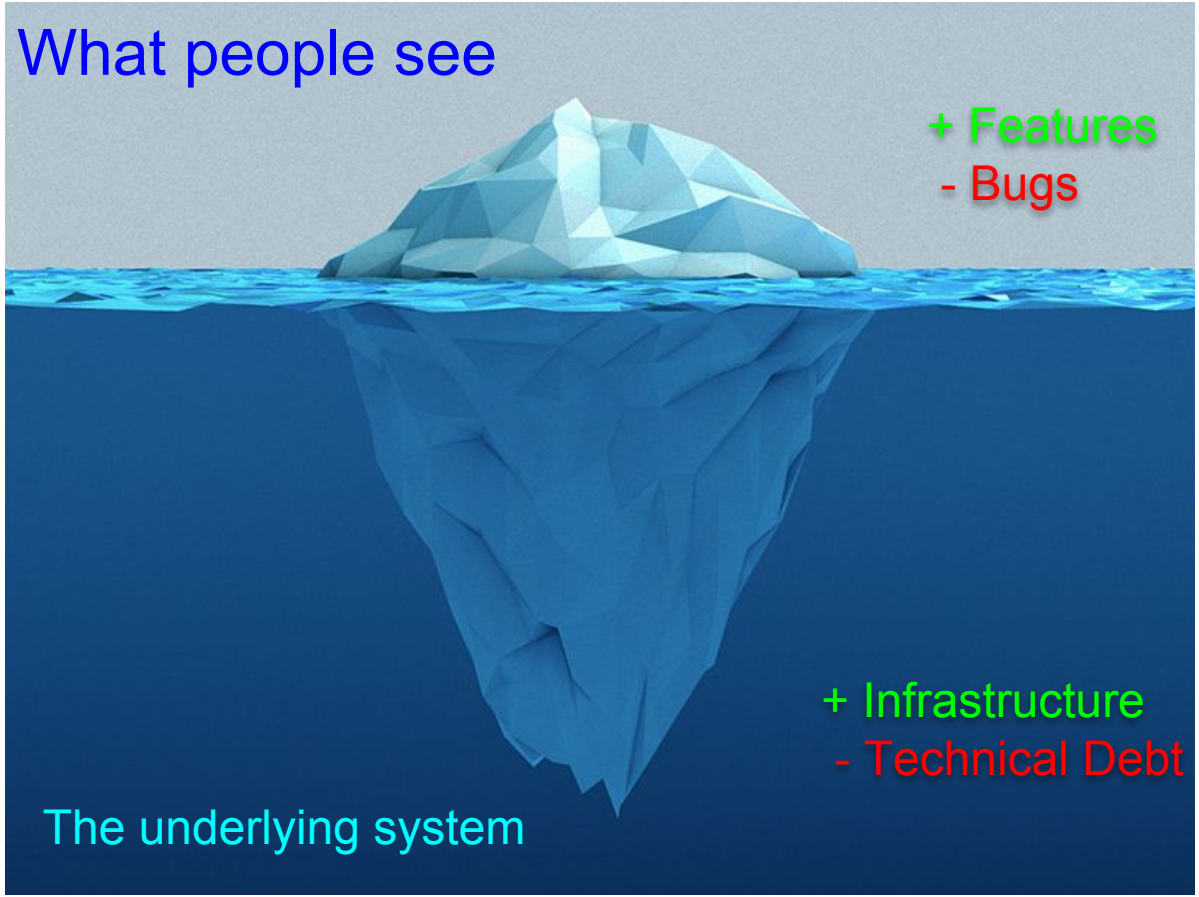
ELMO
CLOUD HR & PAYROLL

Technical Debt

“The road to hell is paved with good intentions”

What is technical debt?

ELMO
CLOUD HR & PAYROLL



Examples of debt

ELMO
CLOUD HR & PAYROLL

- Badly written code
- No tests
- Cache system failing to expire
- Missing validation
- No cache system resulting in slow processes
- Lack of logs & other blind-spots
- Security and stability

What is technical debt?

ELMO
CLOUD HR & PAYROLL

At Best

Accepted Risk

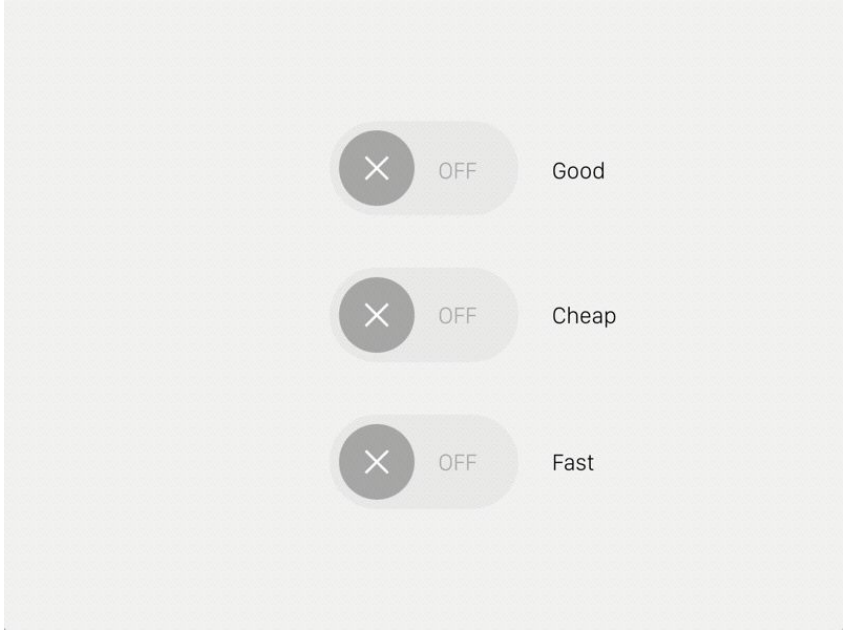
At worst

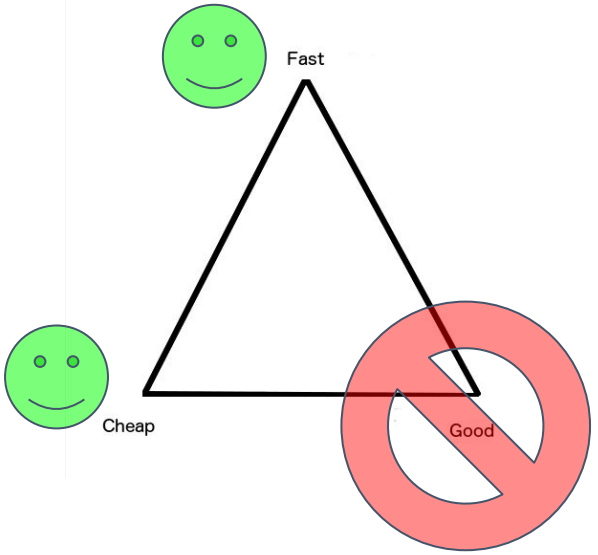
Landmine

Why do we end up with debt?

Sources of debt

Gotta go fast





Sources of debt

Just enough knowledge to be dangerous

- New technology
- New approach
- Under-educated
- Under-researched
- Failure to plan
- Improper prototype
- Etc ...

Sources of debt

Broken windows theory

- If the quality of work is low, there are many “broken windows”
- Leads to more low-quality work
- Leads to loss of motivation
- Normalizes poor quality

https://www.vanschneider.com/loss-in-quality-loss-of-motivation

Avoiding tech debt

Avoiding tech debt

ELMO
CLOUD HR & PAYROLL

Automated testing

ELMO
CLOUD HR & PAYROLL

Automated testing

- Create automation
- Get it green, keep it green
- Keep your coverage high
- You can have multiple automated test systems

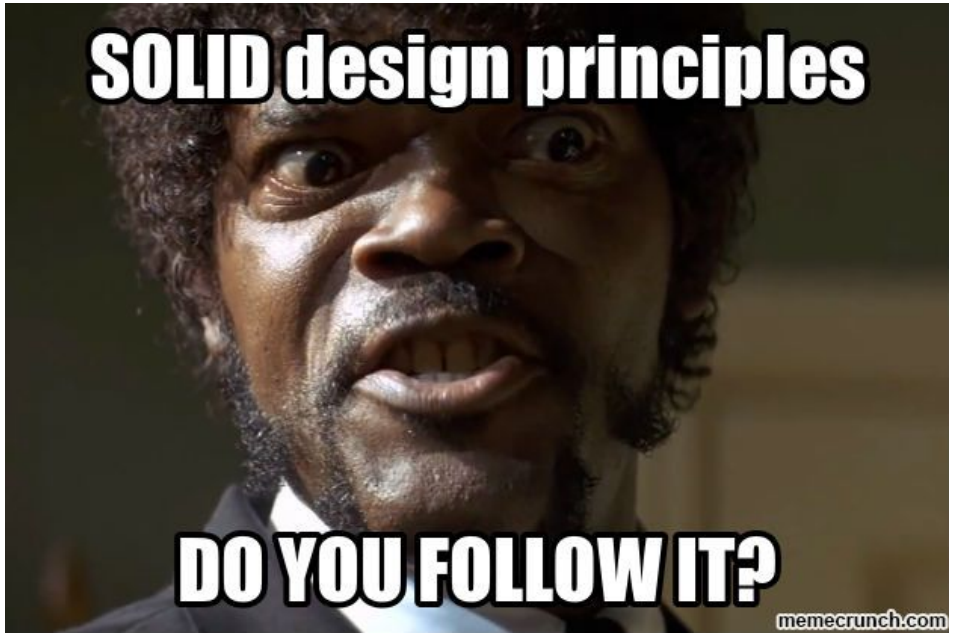
ELMO
CLOUD HR & PAYROLL

Automated testing

- Acceptance end-to-end (E2E) testing
- Functional testing
- Integration testing
- Unit testing
- Linting
- Static Analysis
- Mutation testing

Avoiding tech debt

ELMO
CLOUD HR & PAYROLL



ELMO
CLOUD HR & PAYROLL

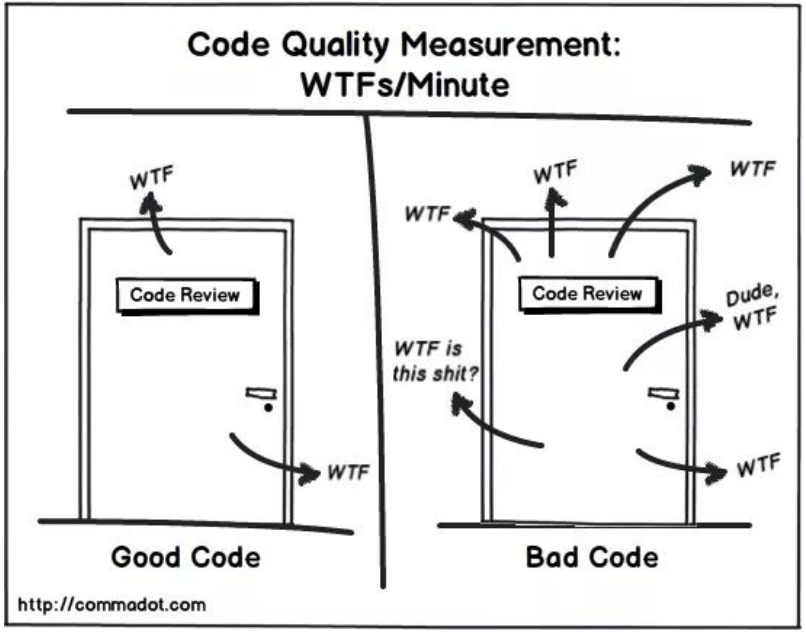
SOLID design principles

SRP	Single Responsibility Principle	Do one thing
OCP	Open/Closed Principle	Don’t make me edit your code
LSP	Liskov Substitution Principle	Children behave like their parents
ISP	Interface Segregation Principle	Make small interfaces for a purpose
DIP	Dependency Inversion Principle	Depend only on abstracts/interfaces

https://medium.com/prod-io/solid-principles-takeaways-ec0825a07247

ELMO
CLOUD HR & PAYROLL

Peer Review



Code = Actual

Comments = Intent

Tests = Proof

Commits = Reason

PR = Summary & Explanation

Docs = Tell others how to use it

Self documenting code isn't.

Safely accumulating debt

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.

```php
class StorageFactory
{
    /**
     * Create storage driver based on selected mode
     *
     * @param ContainerInterface $container
     * @return StorageInterface
     */
    public static function create(ContainerInterface $container): StorageInterface
    {
        $mode = $container->getParameter("storage_mode");

        // Pick storage driver
        // TODO: Well this isn't a good way to do this at all.
        if ($mode == "json") {
            return new JsonStorage($container->getParameter('kernel.root'));
        } elseif ($mode == "redis") {
            return new RedisStorage($container->get('redis'));
        }

        throw new \LogicException("Invalid storage driver specified.");
    }
}
```

Safe

late

```php
class StorageException extends \DomainException
{
    /**
     * Thrown in response to an invalid driver being requested
     */
    public static function invalidDriver(StorageRegistry $registry, string $mode): self
    {
        $names = array_map(function(StorageInterface $driver) {
            return $driver->getName();
        }, $registry->getDrivers());

        return new self(sprintf(
            'Invalid storage driver "%s". Should be one of [%s]',
            $mode,
            implode(', ', $names)
        ));
    }

    /**
     * Thrown when two storage drivers attempt to use the same key
     */
    public static function namingCollision(
      string $name,
      StorageInterface $a,
      StorageInterface $b
    ): self {
        return new self(sprintf(
            'Storage driver collision; two drivers attempted to use the same key "%s" [%s, %s]',
            $name,
            get_class($a),
            get_class($b)
        ));
    }
}
```

```yaml
services:

    my_storage:
        class: MyProject\Storage\StorageInterface
        factory: MyProject\Storage\StorageFactory:create
        arguments:
            - "@my_project.storage.registry"

    my_storage.storage.registry:
        class: MyProject\Storage\StorageRegistry

    my_storage.driver.json:
        class: MyProject\Storage\JsonStorage
        arguments:
            - "%kernel.root%"
        tags:
            - { name: "my_project.storage.driver" }

    my_storage.driver.redis:
        class: MyProject\Storage\RedisStorage
        arguments:
            - "@redis"
        tags:
            - { name: "my_project.storage.driver" }
```

```php
/**
 * Registers all tagged services against the registry
 */
class StorageDriverCompilerPass implements CompilerPassInterface
{
    const TAG = "my_project.storage.driver";
    const SERVICE = "my_storage.storage.registry";
    const SERVICE_METHOD = 'addProcessor';

    /**
     * {@inheritdoc}
     */
    public function process(ContainerBuilder $container)
    {
        $definition = $container->getDefinition(self::SERVICE);

        // Tagged Services: Apply to target service
        $taggedServices = $container->findTaggedServiceIds(self::TAG);
        $taggedServices = array_keys($taggedServices);

        // Add definition method call as arg
        foreach ($taggedServices as $serviceId) {
            $definition->addMethodCall(self::SERVICE_METHOD, [new Reference($serviceId)]);
        }
    }
}
```

```php
    public function getDriver($name) {
        $driver = null;

        if (array_key_exists($name, $this->drivers)) {
          $driver = $this->drivers[$name];
        }

        return $driver;
    }

    /**
     * @return StorageInterface[]
     */
    public function getDrivers(): array {
        return $this->drivers;
    }
}
```

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.
- Avoid over-engineering

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.
- Avoid over-engineering
- Some debt is actually healthy

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.
- Avoid over-engineering
- Some debt is actually healthy
- Gotta go fast!

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

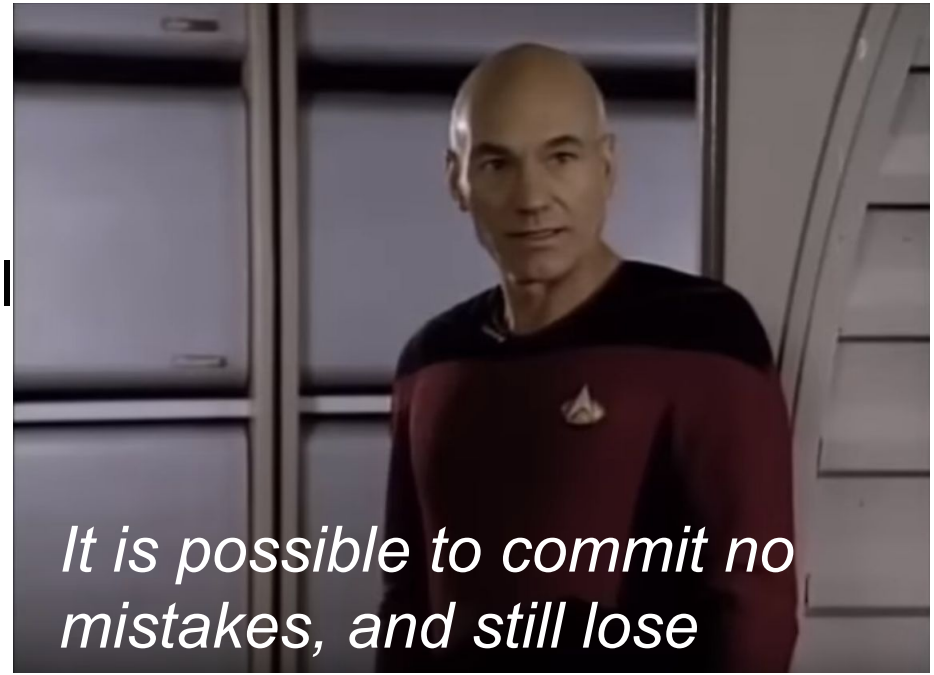
- Lazy.
- Avoid over-engineeri
- Some debt is actually
- Gotta go fast!
- Works for me.

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Why accumulate debt?

- Lazy.
- Avoid over-engineering
- Some debt is actually heal
- Gotta go fast!
- Works for me.
- External forces



Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Making debt safe

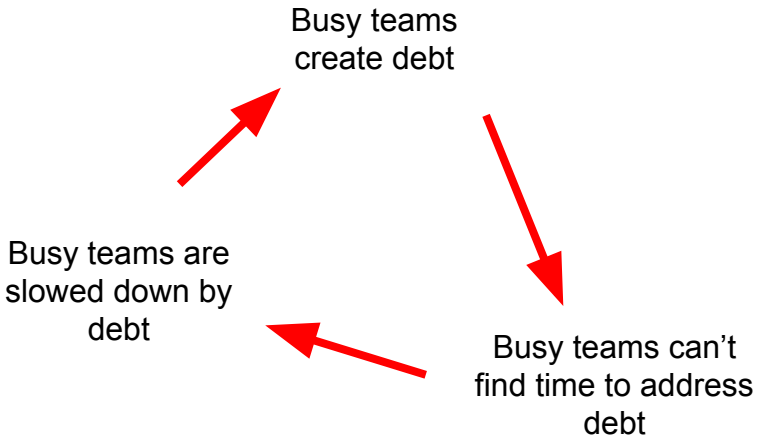
- Quarantine your debt
- Put debt behind clean APIs you (might) keep using
- Hide your evil deeds

Paying back debt

Safely accumulating debt

ELMO
CLOUD HR & PAYROLL

Dedicate time to debt



- Enforce debt be tackled regularly
- Tick / Tock approach
- Add debt to the next sprint
- Track debt time with project velocity


ELMO
CLOUD HR & PAYROLL
Dealing with Technical Debt

Dealing with debt

- **Denial**
  “It's not a bug, it’s a feature.”
- **Anger**
  “I think we should burn it.”
- **Bargaining**
  “If it works, it works ¯_(ツ)_/¯”
- **Depression**
  “Don’t make me touch it, I don’t want to be responsible for it”
- **Acceptance**
  “Hold onto your butts”

Dealing with technical debt

ELMO
CLOUD HR & PAYROLL

Simple Debt

- Add tests
- Refactor code
- Smaller code is easier code!
- Celebrate


Extreme Debt

Extreme Debt

ELMO
CLOUD HR & PAYROLL

- No services
- No domains
- No standards
- No tests
- Lots of users

Extreme Debt

ELMO
CLOUD HR & PAYROLL

BURN IT

Extreme Debt

Dealing with debt

- Denial
- Anger
- Bargaining
- Depression
- Acceptance

Extreme Debt

~~BURN IT~~

Let’s make things *better.*

Extreme Debt

ELMO
CLOUD HR & PAYROLL

Extreme Debt

ELMO
CLOUD HR & PAYROLL

Green-field everything!

- New functions
- New classes
- New libraries
- Microserves

- Clean code
- Few responsibilities
- Lots of tests
- Keep it green!


ELMO
CLOUD HR & PAYROLL
Practical techniques for debt

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Visibility

- Make debt more visible to others
- Jira
- Trello
- Spreadsheets
- Strongly worded emails
- Literally anything

https://www.instagram.com/nathanwpyle/

Extreme Debt

ELMO
CLOUD HR & PAYROLL

Consolidate your debt into easy monthly repayments

- Debt may take multiple passes
- Flag low-hanging fruit with comments
- Small changes clean the way for large changes
- Small changes are better than big bangs

https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Annotations and comments

```php
class MyClass
{

    /**
     * @deprecated User RopeHelper::moreRope instead
     *
     * @param array $props
     * @return string
     */
    public function bitRopey(array $props) {

    }

    /**
     * @refactor/form-security Missing proper validation in forms
     *
     * @param Request $request
     * @param $ropeId
     */
    public function ropeyAction(Request $request, $ropeId)
    {

    }

}
```

ELMO
CLOUD HR & PAYROLL

Test what you can

- Use high-level browser testing to verify your system works.
- High-level tests never lose value.
- Add unit tests for all new code.

Extreme Debt

ELMO
CLOUD HR & PAYROLL

Acceptable risk

- Assume you will break things
- Have a fall-back plan
- DB restores are horrendous
- Code rollback is easy
- Make all changes backwards compatible
- Don’t change behaviors
- Don’t break APIs

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

```php
class ThemeService
{
    /**
     * @param int $id
     * @return array
     */
    public function getTheme($id);

    /**
     * @return array
     */
    public function getThemes();

    /**
     * @param int $id
     * @param string $key
     * @param string $value
     */
    public function setThemeValue($id, $key, $value);
}
```

≠

```php
class ThemeService
{
    /**
     * @param int $id
     * @return Theme
     */
    public function getTheme($id);

    /**
     * @return Theme[]
     */
    public function getThemes();

    /**
     * @param Theme $theme
     */
    public function saveTheme(Theme $theme);
}
```

ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

- Lots of usages
- Bizarro downstream code
- Green fields it!

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

```php
namespace MyProject\AppBundle\Theme;

class ThemeService
{
    /**
     * @param int $id
     * @return array
     */
    public function getTheme($id);

    /**
     * @return array
     */
    public function getThemes();

    /**
     * @param int $id
     * @param string $key
     * @param string $value
     */
    public function setThemeValue($id, $key, $value);
}
```

```php
namespace MyProject\ThemeBundle\Service;

class ThemeService
{
    /**
     * @param int $id
     * @return Theme
     */
    public function getTheme($id);

    /**
     * @return Theme[]
     */
    public function getThemes();

    /**
     * @param Theme $theme
     */
    public function saveTheme(Theme $theme);
}
```

Practical techniques for debt

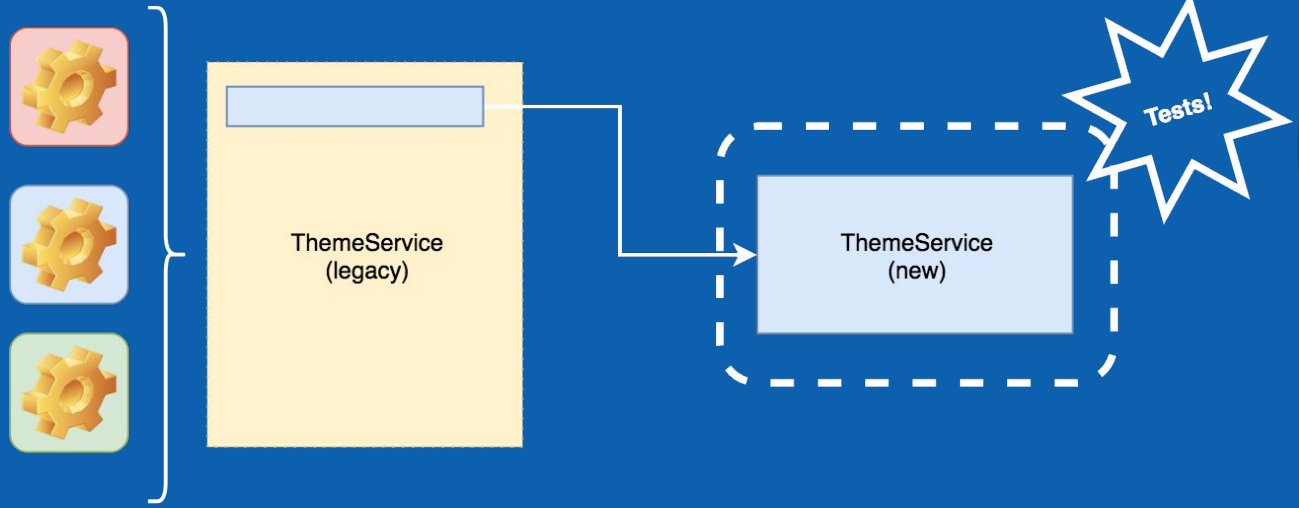
ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

```php
namespace MyProject\AppBundle\Theme;

use MyProject\ThemeBundle\Service\ThemeService as AdaptedThemeService

/**
 * @deprecated Use ThemeBundle\ThemeService instead!
 */
class ThemeService
{
    /**
     * @var AdaptedThemeService
     */
    private $themeService;

    /**
     * @param int $id
     * @return array
     */
    public function getTheme($id)
    {
        return $this->themeService->getTheme($id)->toArray();
    }

}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

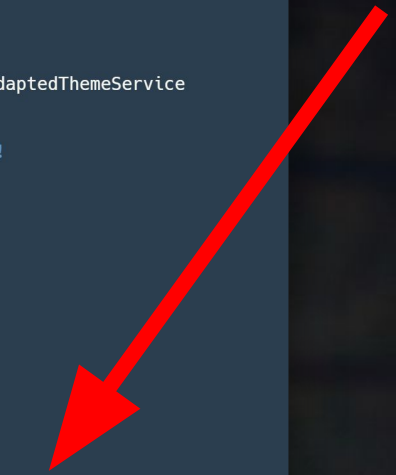


Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

```php
namespace MyProject\AppBundle\Theme;

use MyProject\ThemeBundle\Service\ThemeService as AdaptedThemeService

/**
 * @deprecated Use ThemeBundle\ThemeService instead!
 */
class ThemeService
{
    /**
     * @var AdaptedThemeService
     */
    private $themeService;

    /**
     * @param int $id
     * @return array
     */
    public function getTheme($id)
    {
        return $this->themeService->getTheme($id)->toArray();
    }

}
```



Practical techniques for debt

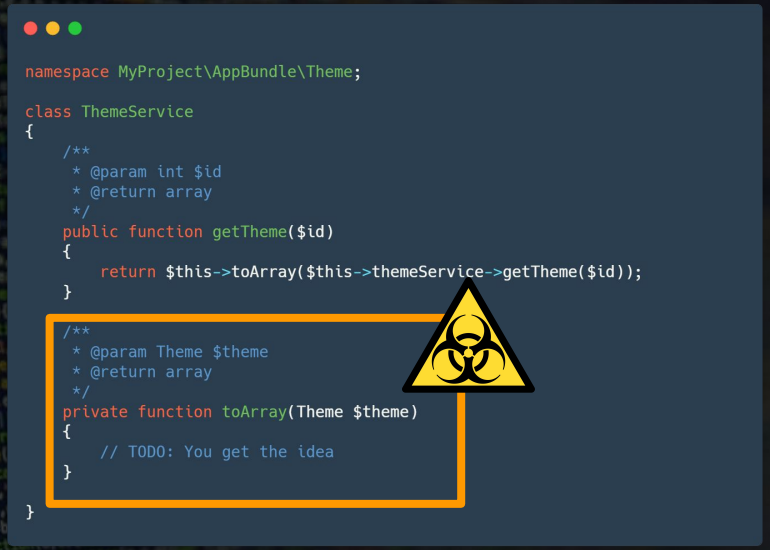
ELMO
CLOUD HR & PAYROLL

Evil Services & Adapters

```php
$theme = $themeService->getTheme();

$theme = $theme->toArray();

$background = $theme['background'];
$logo = $theme['logo'];
$logoUrl = $this->generateLink->image($logo);

$options = (array) $theme['options'];
if (isset($options['menu_sidebar'])) {
    $enable = Menu::SIDEBAR;
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL



```php
namespace MyProject\AppBundle\Theme;

class ThemeService
{
    /**
     * @param int $id
     * @return array
     */
    public function getTheme($id)
    {
        return $this->toArray($this->themeService->getTheme($id));
    }

    /**
     * @param Theme $theme
     * @return array
     */
    private function toArray(Theme $theme)
    {
        // TODO: You get the idea
    }
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Arrays

```php
$theme = $themeService->getTheme(123);

$theme['foreground'];
$theme['background'];
$theme['background2'];
$theme['fontHeader'];
$theme['fontFooter'];
$theme['images']['logo'];
$theme['images']['background'];
$theme['options']['sideAlign']['upper'];

$theme['could']['be']['literally']['anything'];
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Arrays & ArrayAccess

```php
use MyProject\Features\ArrayObjectTrait;

class Theme implements \ArrayAccess
{
    use ArrayObjectTrait;

    private $a;

    private $b;
}
```

```php
/**
 * This type check will throw a type error. Boo!
 */
function getThemeBackground(array $theme)
{
    return $theme['background'];
}
```

```php
/**
 * Allows object properties to be accessed as-if arrays
 */
trait ArrayObjectTrait
{
    public function offsetExists($offset): bool
    {
        return property_exists($this, $offset);
    }

    public function offsetGet($offset)
    {
        return property_exists($this, $offset) ? $this->$offset : null;
    }

    public function offsetSet($offset, $value): void
    {
        if (property_exists($this, $offset)) {
            $this->$offset = $value;
        }
    }

    public function offsetUnset($offset): void
    {
        if (property_exists($this, $offset)) {
            $this->$offset = null;
        }
    }
}
```

Extreme Debt

ELMO
CLOUD HR & PAYROLL

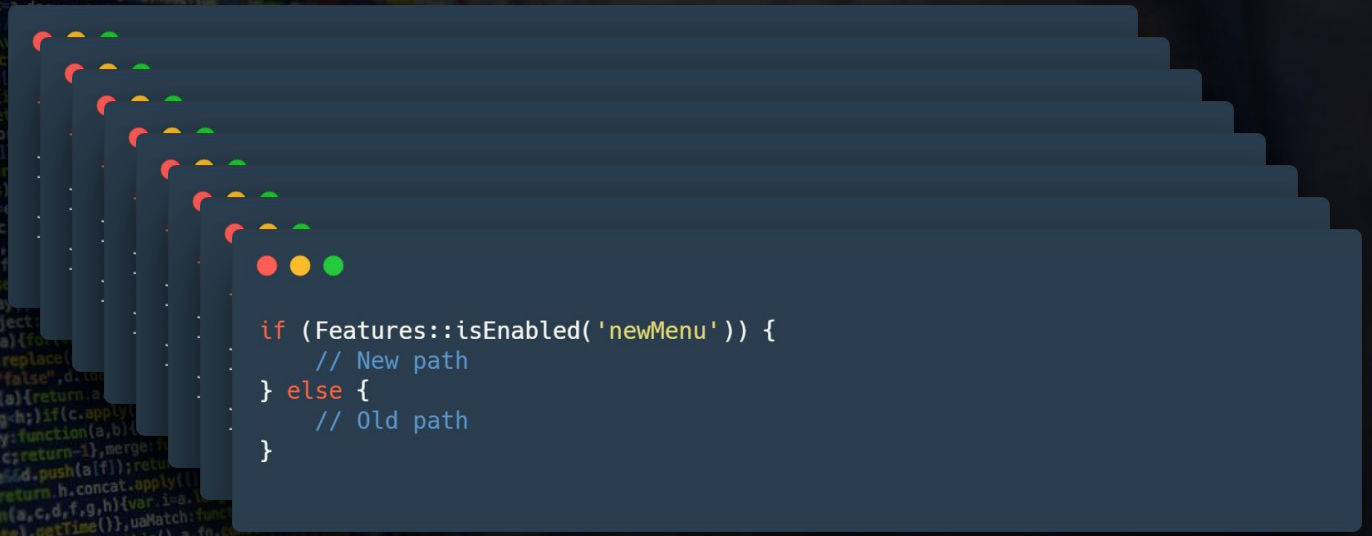
Feature flags

- Hide the line between your features and your releases
- The best flags are the kind you never see

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Feature Flags



Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good Feature Flags & Factories

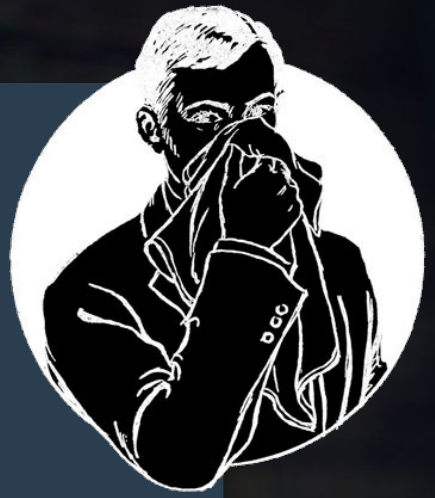
```
class MenuFactory
{
    public function create()
    {
        if (Features::isEnabled('newMenu')) {
            return new HandyDandyManu();
        } else {
            return new LegacyMenu();
        }
    }
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Conditions

```php
if ($object instanceof Alpha) {
    // Do a thing
} elseif ($object instanceof Bravo) {
    // Do another thing
}

switch (get_class($object)) {
    case Alpha::class:
        // Something
        break;
    case Bravo::class:
        // Something else
        break;
}

if ($object->getType() == Alpha::TYPE) {
    // Please stop
} elseif ($object->getType() == Bravo::TYPE) {
    // Oh god why
}
```



Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil Conditions & Abstractions

```php
class AlphaHandler implements HandlerInterface
{
    /** @var Alpha */
    private $model;

    /** @var AlphaService */
    private $alphaService;

    /**
     * @param Alpha $model
     * @param AlphaService $alphaService
     */
    public function __construct(Alpha $model, $alphaService)
    {
        $this->model = $model;
        $this->alphaService = $alphaService;
    }

    public function getName()
    {
        return $model->getPublishedTitle();
    }

    public function delete() {
        $this->alphaService->remove($this->model);
    }
}
```

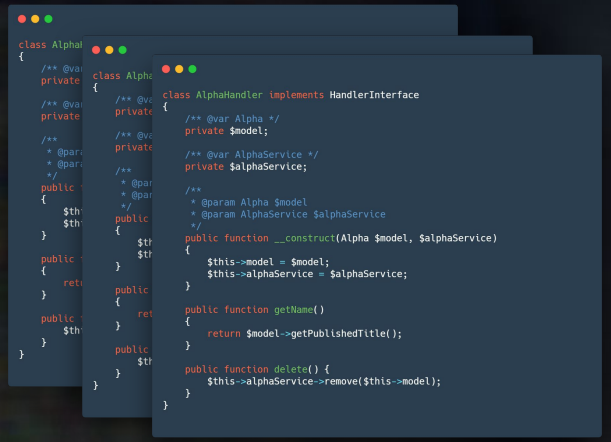
```php
$alpha = $this->getAlpha();

// Wrap it!
$factory = function(Alpha $alpha) {
    return new AlphaHandler($alpha, $this->get('alpha_service'));
};
$handler = $factory($alpha);

// A handled object has a name
$handler->getName();

// Delete it?
if ($request->get('delete')) {
    $handler->delete();
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good Abstraction Factory

```php
class HandlerFactory
{
    private $handlers = [];

    public function registerHandler($factory)
    {
        $this->handlers[] = $factory;
    }

    public function create($object)
    {
        foreach ($this->handlers as $handler) {

            // Use if compatible
            if ($handler->isCompatible($object)) {
                return $handler->createHandler($object);
            }
        }

        return null;
    }
}
```



Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Evil attributes

```php
namespace MyProject\Importer;

interface ImporterInterface
{
    public function getId(): string;

    public function getTitle(): string;
}
```

```php
namespace MyProject\User;

use MyProject\Importer\ImporterInterface;

class UserImporter implements ImporterInterface
{
    public function getId(): string
    {
        return 'user';
    }

    public function getTitle(): string
    {
        return 'my_project.importer.user.title';
    }
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good attributes with Builder pattern

```php
namespace MyProject\User;

use MyProject\Importer\Builder\ImporterBuilder;
use MyProject\Importer\ImporterInterface;

class UserImporter implements ImporterInterface
{
    public function configure(ImporterBuilder $builder): void
    {
        $builder->id('user');
        $builder->name('my_project.importer.user.title');
    }
}
```

```php
namespace MyProject\User;

use MyProject\Importer\ImporterInterface;

class UserImporter implements ImporterInterface
{
    public function getId(): string
    {
        return 'user';
    }

    public function getTitle(): string
    {
        return 'my_project.importer.user.title';
    }
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good attributes with Builder pattern

```php
namespace MyProject\User;

use MyProject\Importer\Builder\ImporterBuilder;
use MyProject\Importer\ImporterInterface;

class UserImporter implements ImporterInterface
{
    public function configure(ImporterBuilder $builder): void
    {
        $builder->id('user');
        $builder->name('my_project.importer.user.title');
    }
}
```

```php
namespace MyProject\Importer;

use MyProject\Importer\Builder\ImporterBuilder;

interface ImporterInterface
{
    public function configure(ImporterBuilder $builder): void;
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good attributes with Builder pattern

```php
namespace MyProject\Importer\Builder;

use MyProject\Importer\ImporterDefinition;

class ImporterBuilder
{
    /** @var ImporterDefinition */
    private $definition;

    public function __construct(ImporterDefinition $definition)
    {
        $this->definition = $definition;
    }

    public function name($name): void {
        $this->definition->title = $name;
    }

    public function id($id): void {
        $this->definition->id = $id;
    }
}
```

```php
namespace MyProject\Importer;

use MyProject\Importer\Builder\ImporterBuilder;

interface ImporterInterface
{
    public function configure(ImporterBuilder $builder): void;
}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good attributes with Builder pattern

```php
namespace MyProject\Importer\Builder;

use MyProject\Importer\ImporterDefinition;

class ImporterBuilder
{
    /** @var ImporterDefinition */
    private $definition;

    public function __construct(ImporterDefinition $definition)
    {
        $this->definition = $definition;
    }

    public function name($name): void {
        $this->definition->title = $name;
    }

    public function id($id): void {
        $this->definition->id = $id;
    }
}
```

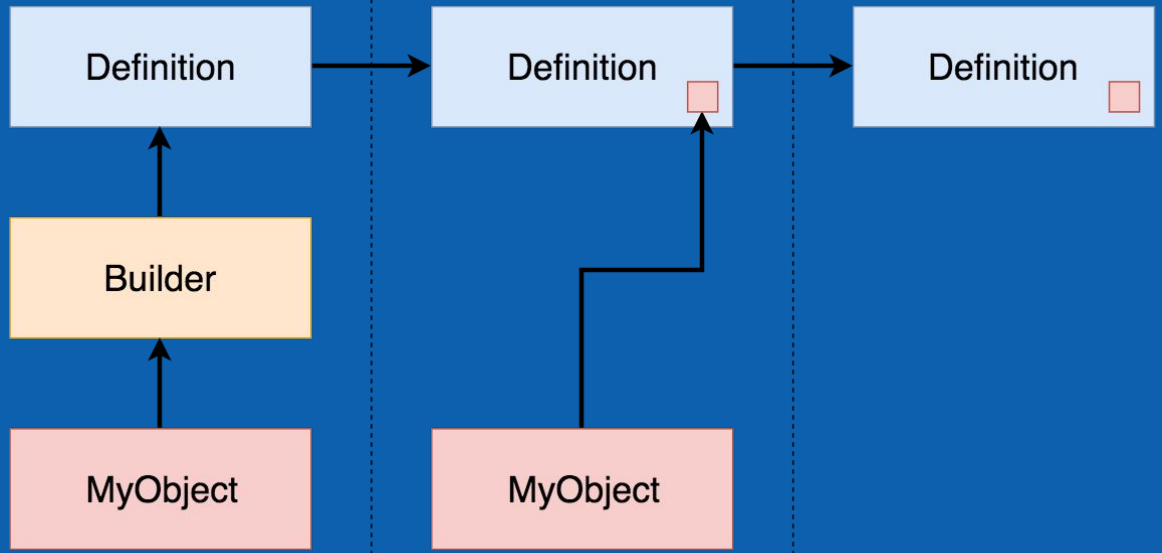
```php
namespace MyProject\Importer;

class ImporterDefinition
{
    /** @var string */
    public $id;

    /** @var string */
    public $title;

    /** @var ImporterInterface */
    public $importer;

}
```

```php
namespace MyProject\Importer;

use MyProject\Importer\Builder\ImporterBuilder;

class ImporterRegistry
{
    /** @var ImporterDefinition[] */
    private $importers = [];

    public function register(ImporterInterface $importer): void
    {
        // Create definition
        $definition = new ImporterDefinition();
        $definition->importer = $importer;

        // Execute builder
        $builder = new ImporterBuilder($definition);
        $importer->configure($builder);

        // Save to collection
        $this->importers[$definition->id] = $definition;
    }

    public function getImporter($id)
    {
        return $this->importers[$id];
    }

}
```

Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Builder pattern layout



Practical techniques for debt

ELMO
CLOUD HR & PAYROLL

Good feature flags with Builder pattern

```php
namespace MyProject\User;

use MyProject\Importer\Builder\ImporterBuilder;
use MyProject\Importer\ImporterInterface;

class UserImporter implements ImporterInterface
{
    public function configure(ImporterBuilder $builder): void
    {
        $builder->id('user');
        $builder->name('my_project_importer_user_title');
        $builder->visible(Feature::isEnabled('user_importer'));
    }
}
```

Final thoughts

ELMO
CLOUD HR & PAYROLL

- Test as much as you can
- Reduce the scope / quarantine your debt
- Figure out the surface area and plan your attack.
- Green-field is always an option
- Some debt is healthy
- Debt is a challenge to be *conquered*

Questions

ELMO
CLOUD HR & PAYROLL

www.elmosoftware.com.au

contactus@elmosoftware.com.au

ELMO Cloud HR & Payroll

ELMO_Software