

PULLING UP YOUR LEGACY APP BY ITS BOOTSTRAPS

by Emily Stamey

<https://joind.in/talk/d4081>



© PHOTO BY MATT STAUFFER



I'm a PHP developer

Organizer of Triangle PHP and
Director of WWCode
Raleigh/Durham

I work at a company that builds
tools for network security threat
assessment

INQUEST



Our Team for this project:

- 3 developers
- 1 project manager
- 1 UX professional



Our applications supported the
College of Engineering, mostly
outside of the classroom

We maintained at least 70 legacy
applications



It's harder to read code than to write it. This is why
code reuse is so hard. - Joel Spolsky

Everybody likes to write reusable code, and no one
wants to reuse anybody else's code.

- Kurt Koppelman (@moonhead)

BOOTSTRAPPING IS IMPORTANT

- We support **legacy** applications, some very old
- Updating **spaghetti** codebases is **RISKY**
- Rewriting large legacy applications is **TIME CONSUMING**
- Bootstrapping meets you in the middle

LEGACY SOFTWARE

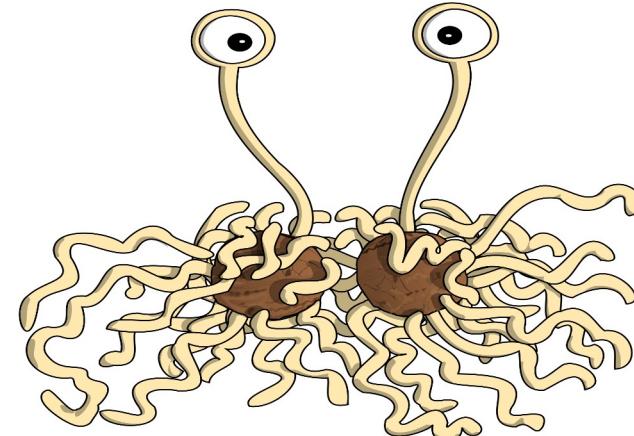
- Software developed using older technologies and practices
- It can be difficult to replace because of its wide use.



- Often a negative term
- Referencing a system as "legacy" often implies that the system is out of date or in need of replacement.

SPAGHETTI CODE

The relationships between pieces of code are so tangled, it's nearly impossible to add or change something without unpredictably breaking something.

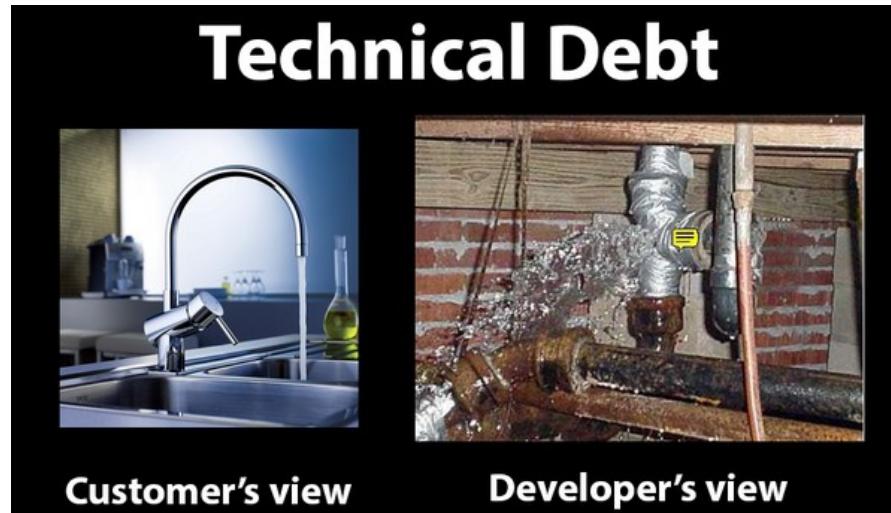


REFACTOR

Technique for restructuring an existing body of code, altering its internal structure **without changing its external behavior.**

TECHNICAL DEBT

A metaphor referring to the eventual consequences of any system design, software architecture or software development within a codebase.



HOW TO MAKE TECHNICAL DEBT

- Leave a codebase untouched
- Develop under a tight deadline
- Let novice programmers build it
- Maintain by multiple developers
- Change goals of the app



BOOTSTRAPPING (SOFTWARE)

**Building onto an existing system
for the purpose of improvement
with the least amount of sweat
equity and development cost in the
process.**



© PHOTO BY @GOATUSERSTORIES

SURVEY YOUR APPLICATION

1. Talk to users of the application
2. Study the codebase
3. Examine the new feature requests

SURVEY YOUR APPLICATION

1. Talk to users of the application
2. Study the codebase
3. Examine the new feature requests

TALK TO USERS

1. Is their process consistent with the application?
2. What are the pain points?
3. Do they have concerns with the application?

Don't rely on developer feedback

USER FEEDBACK: PROCESS

Process was inconsistent with the application

- Language of the customer was different from the application
- Selection Committee used spreadsheet to manage budgets
- Candidates were added to a spreadsheet
(Scholarship Account, Student ID, Amount, Term)

USER FEEDBACK: PAIN POINTS

Process exited and re-entered the system through spreadsheets

High Margin of Error

USER FEEDBACK: CONCERNS

- Many awards were rejected by Financial Aid
- Didn't trust that the best candidates were being chosen
- Scoring algorithm was not clear/effective
- Multiple majors weren't allowed
- **NOT ALL MONEY WAS BEING AWARDED**
- **MONEY LEFT UN-GIVEN ⇒ ANGRY DONORS**

SURVEY YOUR APPLICATION

1. Talk to users of the application
2. Study the codebase
3. Examine the new feature requests

STUDY THE CODEBASE

- Talk to past/current developers
- Verify functionality does what everyone thinks it does
- Look for entanglements

CODEBASE OF SCHOLARSHIPS

- Large App model
 - SQL queries, only slightly dynamic
 - Functions weren't single-purpose
 - No Bounded Contexts between Students, Selection, and Foundation

CODEBASE OF SCHOLARSHIP

- Student application data was a single row in table
 - Academic information wasn't updated when it changed
 - Major was a single column in that row

SURVEY YOUR APPLICATION

1. Study the codebase
2. Talk to users of the application
3. Examine the new feature requests

EXAMINE NEW FEATURE REQUESTS

- What new features are needed?
- How they might be implemented?

NEW FEATURES: SCHOLARSHIPS

- Explicit criteria matching, excluded non-matching applicants
- Current student data, query their GPA, Major, etc at time of selection
- Students have multiple majors

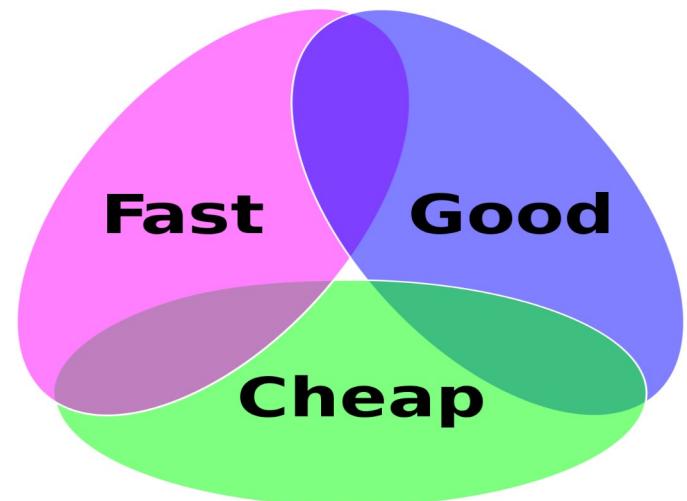
BEFORE YOU START

- Is there another application that can do what it does? Is it better?
- Is this a worthwhile investment?
- If so, what are the Most Valuable Features?



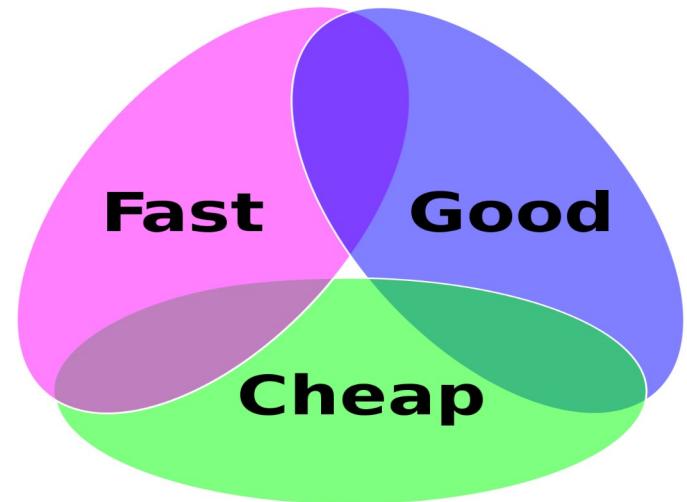
INCLUDE USERS IN DECISIONS

- Explain why this work is necessary
- Be open about errors in the application
- Build trust from the beginning



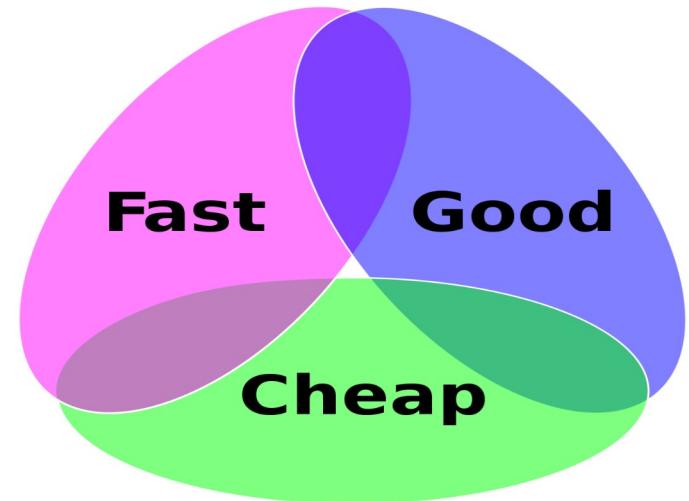
SET EXPECTATIONS

- Customer has an open door to you
 - add work from the side
 - change processes
- Keep the door open anyway!



PRIORITIZE AND SCOPE WORK

- Bootstrapping grows FAST!
 - require more people to be engaged in the process
 - wear people down over time
- Scope the work you are agreeing to do
- Tie the work to the problem it is solving



PLANNING WORK FOR SCHOLARSHIPS

Divided the application based on timeline
we didn't scope the work

MITIGATE RISK

© PHOTO OF NIAMH - ACROBAT

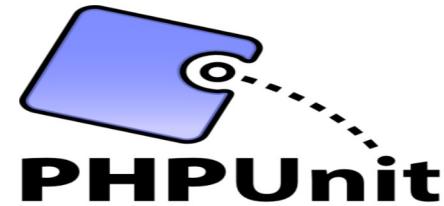
STABILIZE THE CODE

- Version control
 - Stabilize the code base and preserves history
- Development and Staging environments
 - No more developing in production!!!!
 - Created fake student data



TEST EVERYTHING YOU NEED TO KEEP

- Acceptance tests stabilize functionality you need to keep
 - Hooked testing interfaces into framework
 - Used Codeception to view contents of the pages
- Unit and Functional tests for everything you build



COMPOSER

- Allowed us to use libraries
- Checked `composer.json` and `composer.lock` into git
- added `/vendor` to the ` `.gitignore` file



COMPOSER PACKAGES

- Testing with Codeception, PHPUnit, Mockery
- Twig templates
- Pimple container
- Illuminate database (Eloquent)
- Phinx for database migrations



COMPOSER REQUIRE

```
{  
    "name": "itecs/scholarships",  
    "description": "Scholarships application for the College of Engineering.",  
    "require": {  
        "php": ">=5.3.3",  
        "robmorgan/phinx": "*",  
        "pimple/pimple": "~3.0",  
        "ncsu/auth": "dev-master"  
    },  
    "require-dev": {  
        "codeception/codeception": "2.0.*"  
    }  
    ...  
}
```

DATABASE MIGRATIONS: PHINX

- Allows you to change DB across environments
- Gives you power to undo the change if there is a problem

```
$ vendor/bin/phinx create CreateUserLoginsTable
```

```
<?php  
use Phinx\Migration\AbstractMigration;  
  
class CreateUserLoginsTable extends AbstractMigration  
{  
    public function up()  
    {  
        $table = $this->table('users');  
        $table->renameColumn('bio', 'biography');  
    }  
  
    public function down()  
    {  
        $table = $this->table('users');  
        $table->renameColumn('biography', 'bio');  
    }  
}
```

```
$ vendor/bin/phinx migrate
```

```
$ vendor/bin/phinx rollback
```

```
<?php
use Phinx\Migration\AbstractMigration;

class CreateUserLoginsTable extends AbstractMigration
{
    public function change()
    {
        // create the table
        $table = $this->table('user_logins');
        $table->addColumn('user_id', 'integer')
            ->addColumn('created', 'datetime')
            ->create();
    }
}
```

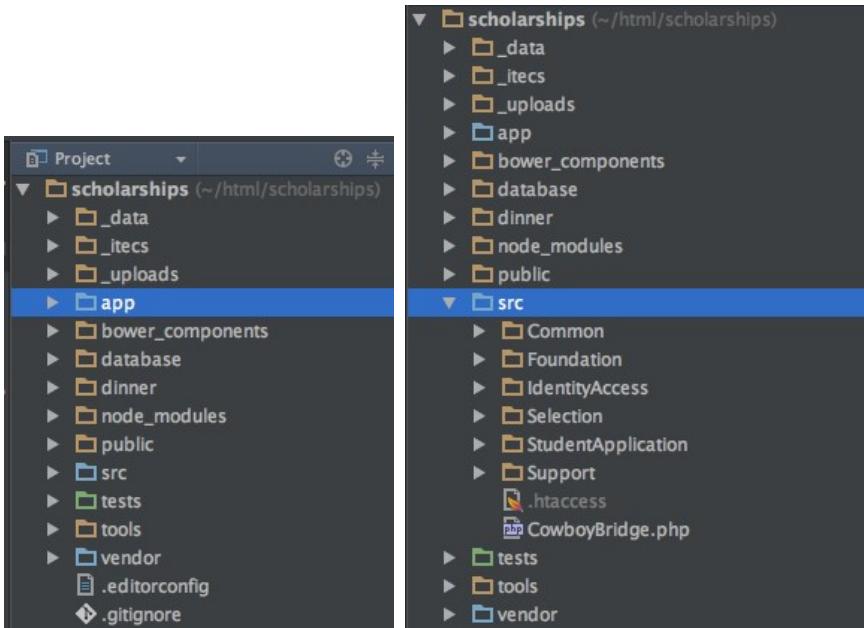
```
$ vendor/bin/phinx migrate
```

```
$ vendor/bin/phinx rollback
```

THE GOOD STUFF

BOOTSTRAPPING: FILETREE

new code in '/src' alongside the '/app' directory



BOOTSTRAPPING: NAMESPACES

- '/src' is given a namespace
- namespaces are autoloaded in composer

```
{  
    "name": "itecs/scholarships",  
    "description": "Scholarships application for the College of Engineering.",  
    ...  
    "autoload": {  
        "psr-4": {  
            "ITECS\\Scholarships\\": [ "src/", "app/core/" ],  
            "Codeception\\Module\\": "src/",  
            "Tests\\Substitute\\": "tests/_helpers/"  
        }  
    }  
}
```

BOOTSTRAPPING: NAMESPACED CLASS

This allows us to reference a class like:

ITECS\Scholarships\Common\Services\GPAService

```
<?php

namespace ITECS\Scholarships\Common\Services;

use ITECS\Scholarships\Common\Values\GPA;
use ITECS\Scholarships\StudentApplication\Domain\Student\StudentId;

interface GpaService
{

    /**
     * @param StudentId $studentId
     *
     * @return GPA
     */
}
```


EVENTS

- Student Submitted Application
- Budget Allocated To Scholarship
- Award Was Given To Student

DEPENDENCY INJECTION

DEPENDENCIES

The dependencies are the objects your class needs to function

GPAService needed a DB connection

INJECT THE DEPENDENCIES

Instantiate the dependency as a parameter in the constructor or use a setter

```
function __construct(Database $database)
{
    $this->database = $database;
}
```

DEPENDENCY INJECTION

- Decouples our code from low level implementation details
- Instantiate our classes with their dependencies
- AND instantiate those dependencies

DEPENDENCY INJECTION CONTAINER

- A map of dependencies your class needs along with the logic to create those dependencies if they haven't been created yet
- Can resolve complex dependencies transparently
- Modular when you need to swap a dependency, only update the container

PIMPLE

- Create your container by instantiating the Pimple class

BASE CONTROLLER

```
<?php

namespace ITECS\Scholarships;

use \ReflectionClass;
use \Log;

/**
 * Base Application Controller Class
 *
 */
class BaseController extends \Controller {

    const BAD_REQUEST=400;
```

'APP/SERVICES.PHP'

```
<?php

use Pimple\Container;
use Illuminate\Database\Capsule\Manager as Capsule;

$container = new Container();

$container['config'] = require_once('config.php');

$container['database'] = function ($c) {
    $capsule = new Capsule;

    // PHP 5.3 doesn't do array de-referencing.
    // @todo update for php54
```

CONFIGURATION FILE

- DB connections
- Base URL
- Set paths to twig templates
- Customize Notice messages
- Set config variables for services

CONFIGURATION FILE

'/app/config.php'

```
<?php
return array(
    /* base url for path in the site */
    'app' => array(
        'base_url' => sprintf('http://localhost:%s/', isset($_SERVER['SERVER_PORT']
            ? $_SERVER['SERVER_PORT'] : ''),
        'index_page' => 'index.php',
        'debug' => FALSE
    ),
    /* DB configuration */
    'db' => array(
        'default' => array(
            'hostname' => "local server"
        )
    )
);
```

We passed this array into a `$container['config']` variable

BOOTSTRAPPING: CONNECTING TO THE FRAMEWORK

- '/app/controllers'
new controllers for the new functionality
- '/app/services.php'
defined and configured twig, database, et al
required in the CI index.php file
- '/app/bindings.php'
the roadmap of our new code and dependencies
required in the '/app/services.php' file

CONTAINER

'/app/bindings.php'

```
<?php

/* example one */

use Scholarships\Selection\Services\IlluminateDatabaseGpaService;

$container['Scholarships\Common\Services\GpaService'] = function($c) {
    return new IlluminateDatabaseGpaService($c['database']);
};
```

CONTAINER

'/app/bindings.php'

```
<?php

/* example two */

use Scholarships\Selection\ApplicantQueryService;

$container['Scholarships\Selection\ApplicantQueryService'] = function($c) {
    return new ApplicantQueryService(
        $c['Scholarships\StudentApplication\StudentQueryService'],
        $c['Scholarships\Common\Services\ResidenciesService'],
        $c['Scholarships\Common\Services\GpaService'],
        $c['Scholarships\Common\Services\UnmetNeedService'],
        $c['Scholarships\Common\Services\CandidateQualificationService']
    );
}
```

CONTROLLERS

```
<?php

class Selectionnext extends BaseController
{
    public function Selectionnext()
    {
        parent::BaseController();

        $this->scholarshipRepository = $this->container['Scholarships\Selection\Sch
        $this->collaborationsService = $this->container['Scholarships\Selection\Col
        $this->committeeService = $this->container['Scholarships\Selection\Committe
        $this->authService = $this->container['Scholarships\IdentityAccess\Authenti
        $this->awardsService = $this->container['Scholarships\Selection\Scholarship
        $this->events = $this->container['Scholarships\Support\Events\EventStore'];


```


SUMMARY: SCHOLARSHIP WINS!

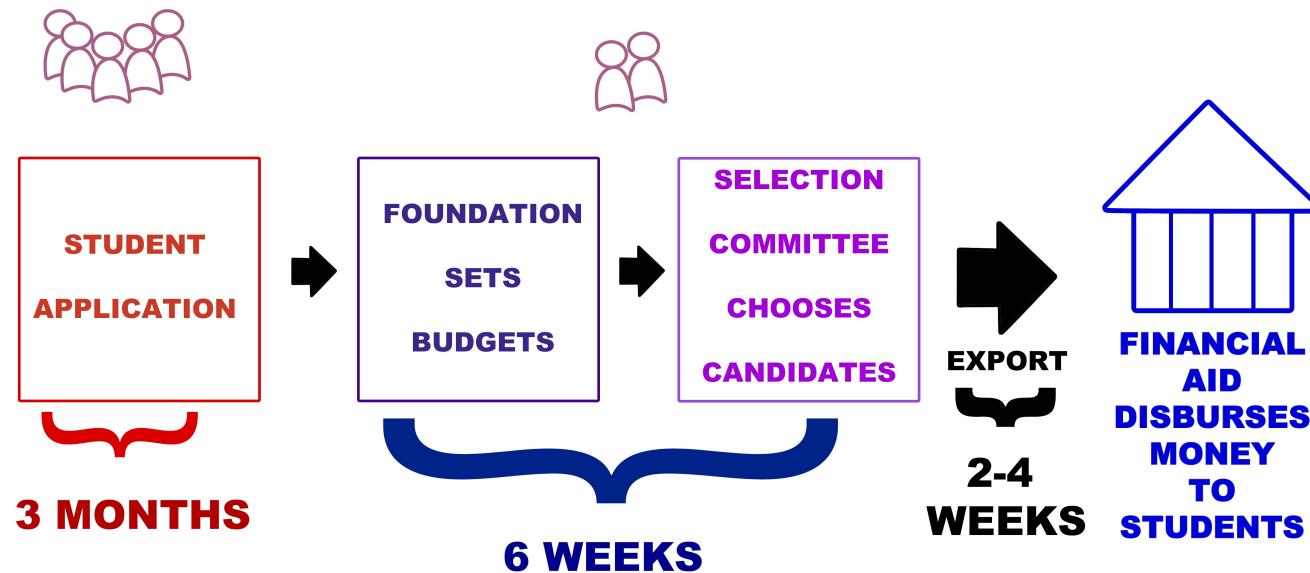
- Implemented multiple majors successfully
- Eliminated unqualified candidates
 - made the scoring easier to read
 - reduced the manual review of candidates
- Selection process was entirely inside the application

SUMMARY: SCHOLARSHIP WINS!

- Restored confidence in selection process!
- Fewer awards were rejected!
- More Scholarship Money was awarded in the application than ever before!

By May 2015: Approximately \$1,074,394 Awarded

SUMMARY: LESSONS LEARNED



Tight deadlines with un-scoped work, we created technical debt that we would have to address in the next academic year

SUMMARY: LESSONS LEARNED

- Bootstrapping Legacy Apps
- Event Sourcing
- Domain-Driven Design
- Command Query Response Segregation
- Project Management
- A LOT!

SUMMARY: ACCOMPLISHMENTS

- Replaced the full Student Application
- Replaced the Selection Process with improved functionality
- Built Event-sourced distribution of scholarship money

OLD SELECTION INTERFACE

Scholarship Committee Portal

Applicants | Scholarships | Engineering Scholarships | Viewing Year: 2016/2017 (current year)

Departmental Scholarships

Choose Department:

192 scholarships for your department.

Show entries Search:

Name	Available Funds	Date Est.	Availability	Endowed	Endowment Date
		11/05/2007			
		01/28/2013	2005		
			2000/2001		
			2000/2001	endowed	
			2000/2001		
		03/24/1992	01/23/2013	endowed	
			11/19/2000		
			01/25/1989		now

NEW SELECTION INTERFACE

College of Engineering Scholarships

Welcome to the selection portion of the scholarship application for College of Engineering students. This screen will give you a quick introduction on how to use our new application.



Award Progress	
Funds Assigned	0%
Total Funds Budget	\$704,974
Assigned Funds	\$864
Remaining Funds	\$704,110

Next Steps - Select a scholarship from the list below to view it in greater detail and assign funds to qualified candidates.

Show 10 entries

Search:

Scholarship	Total Budget	Available Balance
Aldridge, William F.Sr./High Point Sprinkler Syst	\$1,000	\$400
Andy Sepelak Memorial Scholarship	\$10,027	\$10,027
Angel, Robert F. & Romaine S. Endowment	\$6,460	\$6,460

PLANNING: YEAR TWO

- Rollover between Academic years
- The Scholarship CRUD
- Beginning of the year Fund allocations
- Increase/Decrease of Funds

WHAT NOW?

- Backed up and removed all events from the previous year
- Minor code improvements
- Completed the second year's first selection window
- Two years of selection until Academic Works could get up and running
- They awarded over 2 million dollars in Scholarship money using our app
- Academic Works is online.

Recommended reading

- Blog with Bootstrapping details: elstamey.com
- [Paying Technical Debt](#)
- [Dependency Injection](#) - Anthony Ferrara
- [Liquibase](#) - a better migration tool

THANK YOU!

Emily Stamey

Twitter: [@elstamey](https://twitter.com/elstamey)

Joind.in: <https://joind.in/talk/d4081>