

# Refactoring Legacy Code

@sugendran

# Yammer Frontend is a busy place

24 Engineers across 3 offices

5 years of experimentation

Mix of open source and custom frameworks

# Refactoring the Thread List

Core component of Yammer

Had the most amount of experimentation

Hard to understand and slow to work with

Don't hate past team mates

Past decisions were made  
for good reason

# Rewrite vs Refactor

Time to ship is the most important

Many years of changes and bug fixes

Consider the costs to other projects

# Working alone is dangerous

No one works alone at Yammer

On our own we don't have all the answers

A stubborn person will help keep you honest

# Understand the domain

Do you know all the use cases?

Not everything is written down

Limit the scope after the research

# Release early, release often

Get into master as fast as possible

Use a feature gate

Allows the team to be aware of changes



# Fork the tests

Tests give you confidence

Spent the last year getting coverage up

You can't refactor without tests

# Istanbul

<http://gotwarlost.github.io/istanbul/>

## Code coverage report for **feeds/lib/ui/threads/future/thread\_list.js**

Statements: **86.83%** (211 / 243)      Branches: **66.37%** (75 / 113)      Functions: **88.64%** (39 / 44)

[All files](#) » [feeds/lib/ui/threads/future/](#) » [thread\\_list.js](#)

349		},
350		
351		// Fired when a thread is added to the feed
352		handleThreadAdded: function(thread) {
353	61	var index = this.delegate.getModelIndex(thread);
354	61	var component = this._allItems[thread.id];
355		
356	61	if (this.shouldRenderThread(thread, index)) {
357	55	if (this._shouldClearNotices) {
358		this.clearNotices();
359		this._shouldClearNotices = false;
360		}
361	55	if (!component) {
362	38	component = this.createThreadListItem(thread);
363	38	this._allItems[thread.id] = component;
364	38	this._timeUpdater.addComponent(component);
365		} else {
366	17	(component.\$el    component.\$element).detach();
367	17	if (component.setLastSeenMessageId) {
368	17	component.setLastSeenMessageId(this._lastSeenMessage
369		)}
370		}
371	55	component.render();
372	55	this.insertListItemElement(component.\$el    component.\$e
373	55	this.delegate.setThreadAsSeen(thread);
374	6	} else if (this.shouldRenderNewCount(thread, index)) {
375	6	this.showNewItemNotice();
376		}
377		},

# Think about the API

How will others use this component?

Hack up different approaches

Be explicit – don't repeat past mistakes

# Plato

<http://platojs.org>

```
185         item.destroy();
186     }
187 },
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
```

function addMoreButton

Complexity : 1  
Length : 46  
Difficulty : 5.06  
Est # bugs : 0.07

```
addMoreButton: function() {
    var opts = {
        id: 'moreButton',
        clickCallback: _.bind(this
    };
    var button = new Yam.ui.share
    this.add(button, '.yj-list-c
    button.render();
    this._moreButton = button;
```

# Keep the team in the loop

Get feedback before starting

Share the spec and answer questions

Everyone should understand the changes

# When is the project finished?

Don't let scope creep

There will be diminishing returns

Remove the feature gate

# Avoid the long tail of bugs

Release early and have the team find bugs

The AB test system lets us test on Microsoft

Found features we didn't account for

# Future Works?

Write down the things outside of scope

Will there be a phase 2 and 3?

Give other teams a head start



# Before and After

40% less code

Removed asynchronous rendering

Positive feedback from feed projects

# One more Thing

