CodeMash 2019

By Andrew Hinkle



This was my first year at [CodeMash](http://www.codemash.org/) and thankfully the weather behaved unlike the inches of snow I've heard from years past.

Contents

[Conference Overview 2](#_Toc535759359)

[Thursday 01/10/2019 4](#_Toc535759360)

[Practical Steps to Modernizing Legacy Applications by Gregory Beamer 4](#_Toc535759361)

[Leadership Guide for the Reluctant Leader by David Neal 4](#_Toc535759362)

[Hold Up, Wait a Minute, Let Me Put Some Async In it By Matthew Jones 4](#_Toc535759363)

[Learning the Three Types of Microservices by Mike Amundsen 4](#_Toc535759364)

[Does Agile Need Architects by Kevin Fox (ICC – Sponsor Session) 4](#_Toc535759365)

[Becoming an Effective Mentor by Olivia Liddell 4](#_Toc535759366)

[Metaprogramming in .NET by Jason Block 4](#_Toc535759367)

[Friday 01/11/2019 4](#_Toc535759368)

[DISCovering your team: A Guide to Effective Communication by Patrick Badley and Jared Faris 4](#_Toc535759369)

[Versioning 1.0.1 by Jon Skeet 4](#_Toc535759370)

[Data management in a Microservices world by Gerald Venzl 4](#_Toc535759371)

[Patterns and Architectures Beyond Microservices by Stephan Shary 4](#_Toc535759372)

[How to Improve Your Task Estimates by 0% to 1,000% by Dave Sweeton (Stout Systems – Sponsor Session) 4](#_Toc535759373)

[The Two Question Code Quiz: How to Interview Programmers Effectively by Scott Connerly 4](#_Toc535759374)

[If Your Tests Could Talk by James Balmert 4](#_Toc535759375)

[Conclusion 4](#_Toc535759376)

# Conference Overview

The first 2 days of Pre-Compiler sessions have pre-requisites that require you to bring your own device (laptop/tablet/etc.), download/install apps and source code where you'll get hands on workshops and training. JD always attends these sessions (2019, [2018](https://www.danylkoweb.com/Blog/codemash-v2018-LR), [2017](https://www.danylkoweb.com/blog/codemash-2017-new-friends-old-friends-great-times-IS), [2016](https://www.danylkoweb.com/Blog/codemash-2016-my-exhaustive-week-DP), and [2015](https://www.danylkoweb.com/Blog/how-i-used-codemash-2015-to-determine-my-technology-plans-for-2015-97)), so I'll let him cover them.

The last 2 days consist of standard 1 hour sessions with a 15 minute break in-between. On Thursday it started at 8:00AM EST and ended at 5:45PM EST. Friday started at 8:30AM EST and ended at 5:00PM EST with closing remarks, thanks to staff and presenters, and sponsor giveaways finished before 6:00PM EST.

There was no Keynote sessions which is fine as it gave everyone time for an additional session of their choice. CodeMash 2019 had 8 sessions on Thursday and 7 sessions on Friday. That's definitely more than any conferences I've attended to date. There were roughly a dozen sessions available at each time slot, so there were plenty of options. The conference doesn't target a specific technology. This led me to choose soft skills in some time slots which I've had a hard time justifying to myself in the past. One attendee, a developer, told me he almost exclusively chooses the soft skill sessions to work on personal growth, because the sessions here are that good. He was right, they were good.

The [Kalahari Resorts & Conventions in Sandusky, OH](https://www.kalahariresorts.com/ohio/things-to-do/weddings-and-meetings/meetings-and-conventions/) was an awesome venue for CodeMash 2019. The meals were wonderful and worth the time to eat them. There was plenty of room to walk between sessions while still supporting sponsor booths in the hallways which is saying a lot for the roughly 2500 attendees present. The audio and video in each conference room was excellent. Oddly enough, the only complaint I heard from the main suite were the bright stage lights irritating the guests at my table. We may have just been in the anti-sweet spot for the lights. The main suite is huge with multiple screens making it pretty easy to see what was going on even from the back of the room.

Overall, the only thing I heard negative about the conference is the choice of lunch or session. Whether you left a session early, got to one late, skipped a session or lunch, something doesn't feel right here in the pit of my stomach. Yeah, I went there. I missed one session on Thursday for this very reason.

I saw many people in the main suite eating, skipping the session. I wonder how much better the session attendance would be if the conference set aside a full half hour for lunch instead of the 15 minutes to run between sessions. They could have a 15 minute sponsor presentation while attendees ate. This could also give us more time to speak with the sponsors in the hallways.

The 2-day conference cost of $350 is more than other conferences I've attended, but comes with more amenities such as access to the Kalahari waterpark, breakfast, lunch, (Thursday night dinner), credit towards drinks and snacks, game nights, and more. The Kalahari is a wonderful hotel, but if it's a bit more than what you want, there are multiple good hotels nearby, such as the [Great Wolf Lodge Sandusky, OH](https://www.greatwolf.com/sandusky) which is where I stayed with my family. CodeMash even provided shuttle service between several of the hotels adding a nice convenience.

Overall, CodeMash 2019 was an awe inspiring conference with a great location and sessions. It's considered a National conference where people come from all over the nation and other countries. It's a reunion of friends and colleagues. If you've never been before I recommend trying it out.

# Slideshows and Code Repositories

I dread looking for slideshows and code repositories after a conference. Some are included in the presentation or they are promised to be linked in twitter or slack. Unfortunately they are never in one place where session goers may find them quickly. I get the extra maintenance required to post this information on the conference's web site or a simple github repository that just contains a spreadsheet as an example with the {year}.{conference}.{xlsx}. Thankfully, while searching on [Twitter #CodeMash](https://twitter.com/hashtag/CodeMash?src=hash) I found that [@ikeif](https://twitter.com/ikeif) compiled a list of [CodeMash 2019 Presentations](https://docs.google.com/spreadsheets/d/1omYTaGvLAbkJ-AGphM-WO90iE5O5s5-1dG2srKbNFYo/edit#gid=0) with links in a nice spreadsheet for us. Thanks!

The last session's slideshow was created in Keynote (.key) which is an Apple program and therefore is only available for Apple Mac and iDevices only, there is no Windows version. I've never had troubles opening slideshows in the past, since they were saved in generic Microsoft formats or in slideshow sites. Something I hadn't considered in the past, but presenters should save their slideshows in a format that users can open without much effort or they won't bother trying. The last session stored the presentation as a .key file.

# Thursday 01/10/2019

## Practical Steps to Modernizing Legacy Applications by Gregory Beamer

[Slideshow](https://github.com/gbworld/LegacyModernization)

## Leadership Guide for the Reluctant Leader by David Neal

[Slideshow](https://speakerdeck.com/reverentgeek/leadership-guide-for-the-reluctant-leader-ndc-sydney-2018)

## Hold Up, Wait a Minute, Let Me Put Some Async In it By Matthew Jones

[Slideshow](https://exceptionnotfound.net/downloads/slides/CodeMash2019/LetMePutSomeAsyncInIt.pptx)

## Learning the Three Types of Microservices by Mike Amundsen

[Slideshow](http://amundsen.com/talks/2019-01-codemash/2019-01-codemash-three-types-msc.pdf), [Webpage](http://amundsen.com/talks/2019-01-codemash/index.html)

## Does Agile Need Architects by Kevin Fox (ICC – Sponsor Session)

No slideshow links at the time of writing.

## Becoming an Effective Mentor by Olivia Liddell

[Slideshow](https://www.slideshare.net/OliviaLiddell/olivia-liddell-codemash-2019-becoming-an-effective-mentor)

## Metaprogramming in .NET by Jason Block

[Slideshow](https://github.com/JasonBock/Presentations/blob/master/Metaprogramming%20in%20.NET.pptx)

# Friday 01/11/2019

## DISCovering your team: A Guide to Effective Communication by Patrick Badley and Jared Faris

[Slideshow](https://www.slideshare.net/PatrickBadley/discovering-your-team-codemash)

## Versioning 1.0.1 by Jon Skeet

No slideshow links at the time of writing.

I'll give a shout out to [Gene Gotimer](https://twitter.com/CoverosGene) for tweeting out some great [versioning notes](https://twitter.com/CoverosGene/status/1083737167518711811) on this session.

## Data management in a Microservices world by Gerald Venzl

[Slideshow](https://www.slideshare.net/gvenzl/data-management-in-a-microservices-world-128345289)

## Patterns and Architectures Beyond Microservices by Stephan Shary

No slideshow links at the time of writing.

## How to Improve Your Task Estimates by 0% to 1,000% by Dave Sweeton (Stout Systems – Sponsor Session)

[Slideshow](https://drive.google.com/file/d/14qlSVn0jQqMb7csvs_pYV-hEPzIONWnl/view)

## The Two Question Code Quiz: How to Interview Programmers Effectively by Scott Connerly

[Slideshow](https://slides.com/scottconnerly/2questioncodequiz)

## If Your Tests Could Talk by James Balmert

[Slideshow](https://github.com/jbalmert/if_your_tests_could_talk) (It is a ".key" file)

The Classic [Test-Driven Development (TDD)](https://technologyconversations.com/2014/09/30/test-driven-development-tdd/) follows the pattern of red/green/refactor that eventually breaks up the responsibilities into separate classes. Here's an [example walkthrough](https://technologyconversations.com/2013/12/20/test-driven-development-tdd-example-walkthrough/) written in Java that even C# developers like myself can follow.

Jimmy promoted the Mockist TDD variation during this session. He created a tic-tac-toe game and defined the first scenario for a simple win with 3 X's in a row. He defined the classes he thought he would need such as players, game board, and a game evaluator to detect when the game was over. He created the classes and stubbed methods that he could then mock and inject into the other classes.

After each refactoring he would review his work and record feedback by evaluating the following three types. Most of this feedback falls back to following the [SOLID](https://en.wikipedia.org/wiki/SOLID) principles.

1. Abstraction: ex. Is this class exposing internal state?
2. Responsibility: ex. Too many responsibilities?
3. Usability: ex. Is it easy to understand what this object does?

The thought exercise was an intriguing point of view. After reviewing business requirements for a project and deciding between new or updating an existing system, I usually have a good idea of the objects I want to create similar to this approach. As I create each object, I'll test just the logic in it. If there are any dependencies, I'll inject mocks of them through the constructor and verify that they were called when expected, with the expected values, and have them return an expected result necessary for the object I'm testing.

I'm not interested in testing the behaviors of the dependencies, just that they were called. The only fallback I ever run into with my variation is that sometimes integration may be missed, so at some point an actual integration test will need to be performed, but that's not a unit test.

Overall Jimmy did a fine job going through all of the feedback loops, explaining the reasons for each change, and performing the refactoring. There was some repetition, but it was reasonable considering we are talking about TDD and sometimes you start a refactoring only to realize it didn't work and go back and reevaluate. That just reinforced the need for quicker feedback.

Additional References:

1. [Mocks Aren't Stubs by Martin Fowler](https://martinfowler.com/articles/mocksArentStubs.html#ClassicalAndMockistTesting) which includes a discussion on Classical and Mockist Testing
2. [Should I practice "mockist" or "classical" TDD? - StackOverflow](https://stackoverflow.com/questions/184666/should-i-practice-mockist-or-classical-tdd)

# Conclusion