Benefits of Test Driven Development

Venkat Subramaniam
venkats@agiledeveloper.com
@venkat_s

Why do it?

Programming already is time consuming, why would you take on additional task of writing tests?

Why do it?

This one is counterintuitive—to save time!

How?

Let's answer that question...

What's Agility?

It's our ability, as individuals, teams, and organizations to respond to ever changing business conditions

What's Agile Development?

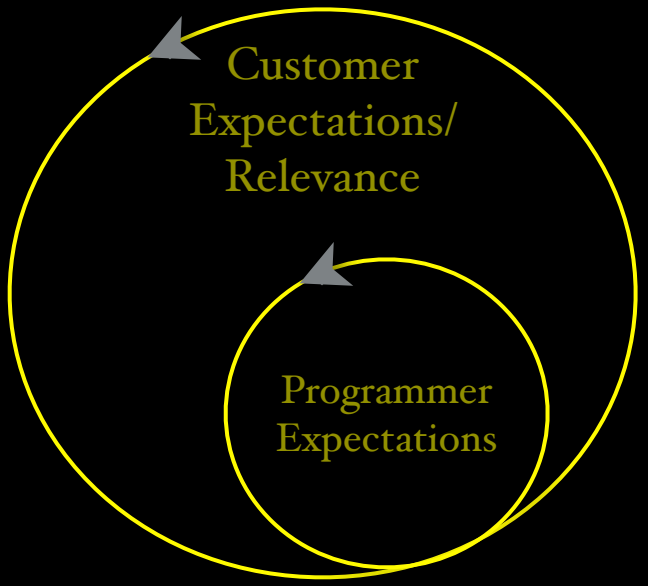
It is Feedback Driven Development

Feedback is Critical

The requirements you start with may not be the ones you end up within the release

If your objective is to build what your customers wanted, you will fail—you need to build what they still want, what's relevant

Feedback Driven



Which is more important?

Feedback Driven

Both types of feedback are equally important

You want rapid feedback
You want to avoid Whack-a-mole software

Sustainability is the key

Agile development is not about running fast...

It is about running fast in the right direction at a sustainable pace

TDD helps?

TDD is a way to get rapid feedback

As you evolve the system based on feedback, bug fixes, and additional features, it tells you...

...that the code worked and continues to work as expected

But it costs to do TDD

Dr. Laurie William's research study...

... experienced programmers

... two teams

... one doing TDD, the other simply coding

...result?

But it costs to do TDD

In the beginning, first version

team that did TDD took 16% more time

When asked to make change, team that did TDD came back first and things simply worked

Team that did not do TDD not only came back much later, but their code no longer worked... they had created whack-a-mole software!

Cost of not doing it?

We know this as a pain in the industry

delayed software release

hard to maintain

hard to predict

hard to change

frustration for everyone, programmers, testers, managers, marketing, sales, support, ...the entire organization

Yet, industry-wide...

"Insanity: Doing the same thing over and over again and expecting different result"—Einstein

Software and Complexity

One of the most troubling words in software development is "entropy," knowingly or not, we spend lots of effort dealing with that.

Software Rot

Software gets complex rather quickly and becomes unmanageable

Several forces are against us

the domain

the infrastructure

the tools

and the change

How to cope?

Two kinds of design: Strategic and Tactical

Strategic design is important, but we have to do it to the right level

Keep it minimal and evolve

TDD helps with tactical design, to keep the complexity at manageable level, continuously

Automation is critical

Without the proper amount of automation, it is impossible to sustain software development - agile or not

What Pushes us to Waterfall?

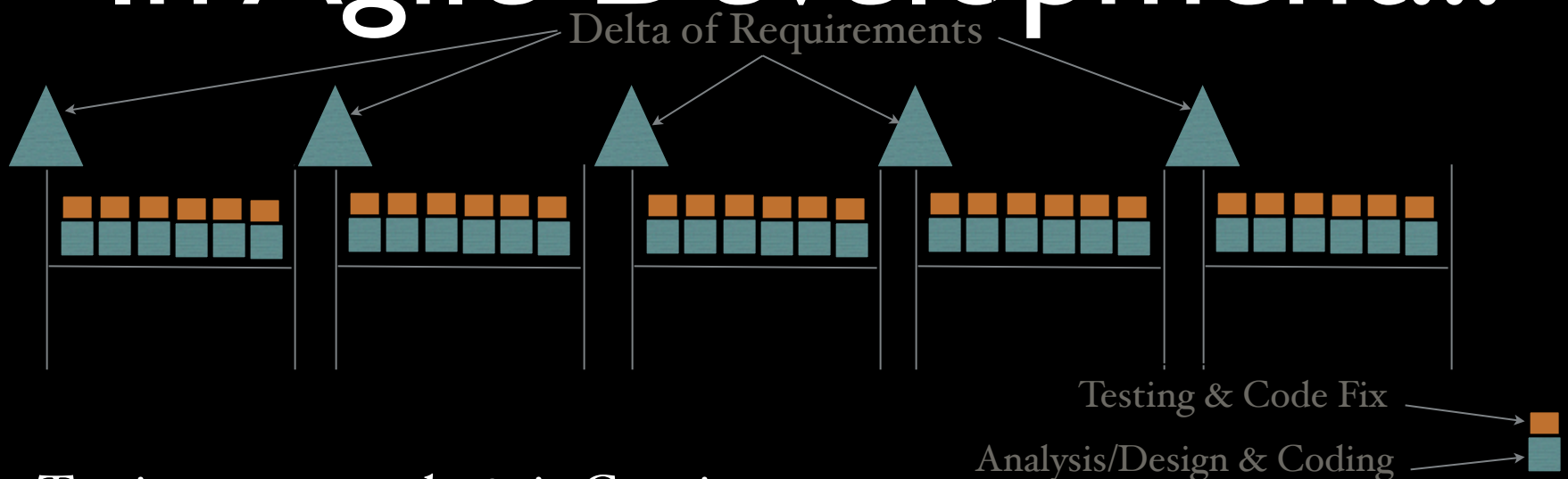
Too late in the game

Often pressure to release

QA become defenders

Often looked at as adversaries

In Agile Development...



Testing starts early & is Continuous

Don't wait until end of iteration to test–test frequently and regularly

Application is exercised constantly, no surprises later

QA become support

Not adversaries, become part of the team

Work with customer and programmers—co-located with them

Tenets of Testing

As a tester, your responsibility is to author tests, not run them!

Automation is Essential

"Error rate in manual testing is comparable to the bug rate in code being tested."—Boriz Beizer

You can't sustain the pace with manual testing

Starting very early with automation makes life easy

Types of Tests

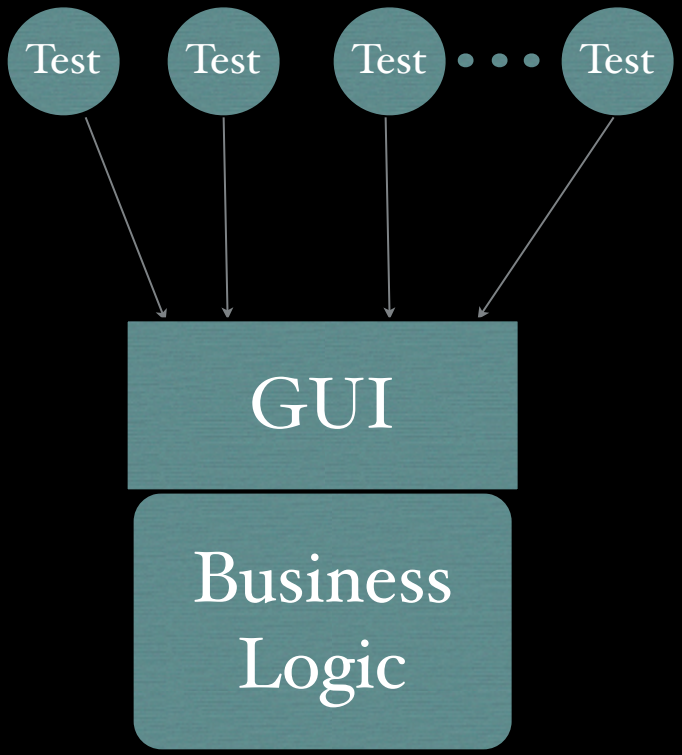
White-box testing—Programmers

Test greatly influences and is influenced by the internal design of the code. Is fine grained.

Black-box testing—Testers and customers with collaboration with Programmers

Test focuses on the external behavior of the code without regard to its internal structure. Is coarse grained.

Where not to Automate?

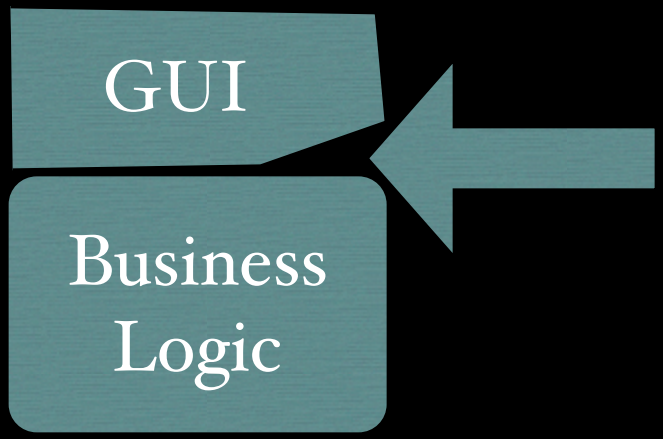


Very hard to automate

Not Effective

Too Brittle

Where to Automate?



Run Automated Tests
one level below GUI

What's Needed for TDD?

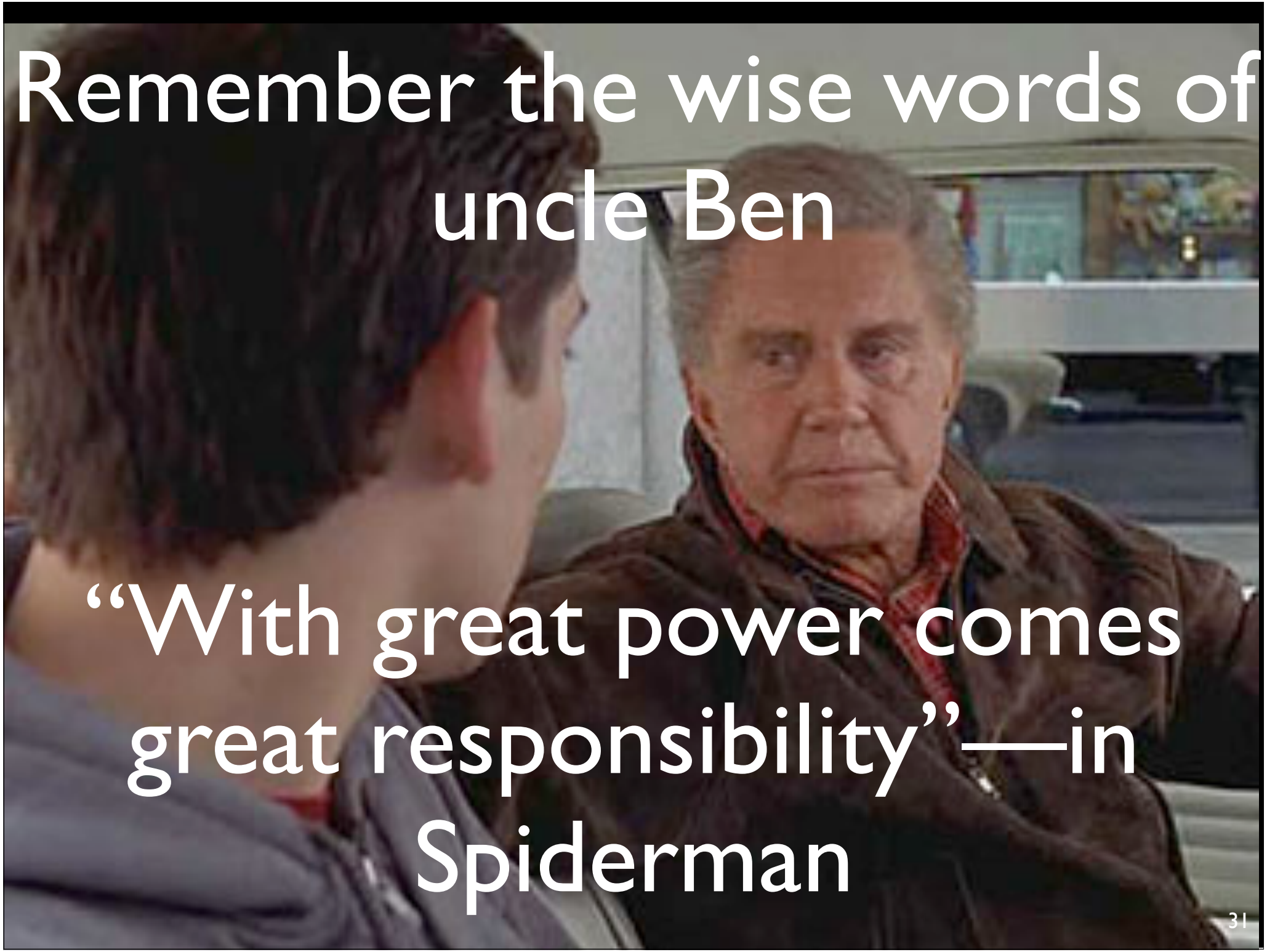
One word: Discipline

Everything else follows it

Programming without automated testing is a roadkill

Unit testing is the software equivalent of exercising

"Some people change when they see the light, others when they feel the heat"—Caroline Schoeder.


Remember the wise words of uncle Ben
"With great power comes great responsibility"—in Spiderman

Venkat Subramaniam
venkats@agiledeveloper.com
twitter: @venkat_s

Thank You!